CHAPTER 9

Database and Information Management

Learning Objectives

- Explain how databases work and identify their basic structural elements
- Identify some common information systems that use databases
- Describe the elements of a database management system (DBMS) and their functions
- Discuss the different database classifications
- Explain how databases are designed
- Differentiate among methods of data processing
- Describe the areas of responsibility for a database administrator

CyberScenario

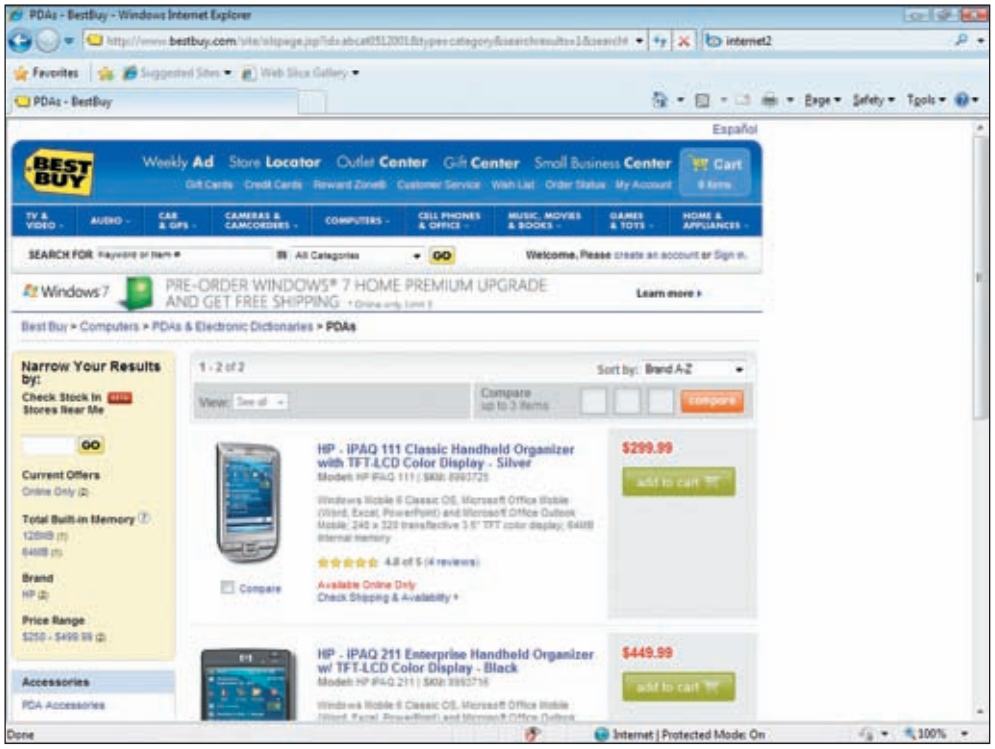
FATHER'S DAY WAS FAST APPROACHING and Lenora still couldn't decide what kind of gift to buy her father-in-law. She was never really sure if he appreciated the cologne, neckties, or fishing gear she usually ended up buying. Hoping that he had taken her suggestion at Christmas to set up an online wish list, Lenora began to check some of the more popular wish list and gift registry services on the Internet.

Finding his wish list on her first try, Lenora began to peruse the items he selected. The range of choices was impressive, from books to camping supplies to artwork. She narrowed down the list by selecting her preferred price range and then began to read product reviews from other consumers. She then checked for recently purchased items and was presented with a hint to select a leather case and software for the GPS unit that someone else bought for him this week. Lenora smiled to herself as she pictured his surprise at her buying a useful birthday gift!

Next, she chose wrapping paper, customized a birthday card, and clicked the *Buy* button. The Web site informed Lenora of the total cost, and confirmed that the gifts would arrive on time. The entire shopping experience had taken a mere 12 minutes. Lenora breathed a sigh of relief, knowing that this time she had bought her father-in-law gifts he was sure to like. Lenora's search illustrates how e-commerce transactions involve a series of online database interactions. The initial search for a product queries (asks) a database for information. A sales record is created in the company's database when a payment is made. That database then uses the customer's credit card number to access a bank database and electronically transfer funds to the retailer. Finally, the e-commerce site's database interacts with a shipping company's database to ensure that the purchase reaches the customer without delay.

Database Basics

A **database** is a computerized system for storing information in an organized manner so that it can be searched for and retrieved when needed. Businesses, government groups, private organizations, and academic institutions all use databases, and they represent the dominant use of computing power in the business world



Electronic-commerce sites use databases to record price, sales, and other information about their products.

today. Without databases the Internal Revenue Service could not collect income taxes, the American Red Cross could not allocate funds, and colleges across the country could not operate efficiently. Schools use databases to store and handle grades, class schedules, tuition payments, library fines, and other records relevant to education. Colleges typically use enrollment trend reports to plan for new courses and to branch out into new academic areas. In these cases they might use not only their own database information, but also reports from national college associations and the federal government.

Databases are used for more than routine operations. Executives commonly consult company databases to aid in their decision making. If a company is considering opening a new branch office, managers and other strategic planners can carefully examine databases describing the purchasing habits of local consumers in order to choose the best location.

Data vs. Information

The terms *data* and *information* are key concepts in understanding the importance of computerized databases. Recall from Chapter 1 that data is a collection of raw, unorganized (unprocessed) content in the form of words, numbers, sounds, or images. Data associated with other useful data on the same topic becomes information. The ability to associate or organize stored data in a variety of meaningful ways represents the power of database software. For example, an insurance company might store ZIP code information in its database, but a ZIP code alone is not particularly useful. However, when the ZIP code is associated with other data such as a name, address, or phone number, this set of organized data becomes information that can be used to locate and identify a customer.

Historical Database Forms

Databases as storage systems existed long before computers came into being. Important records such as birth certificates, medical histories, income tax files, payroll records, and car license data were stored on paper before the first database software was developed in the 1950s and 1960s. These printed documents were usually collected and organized in filing cabinets. Locating information and structuring it for various purposes was possible, but it involved sifting through stacks of paper, identifying the desired data, and then writing or typing selected items on more paper so that people could analyze the information. For example, if a health department wanted to determine if a flu outbreak had resulted in a significant increase in deaths within a certain age group, workers had to read mounds of death records and manually note the causes of death and the ages of the victims. Today the same task could be executed in minutes using a computerized database.

Computerized databases have become so useful in all areas of public life that paper-based storage systems are rarely encountered. The few exceptions are small independent medical offices and some small retail shops, but even these businesses are rapidly moving to electronic systems.

INTERNET RESOURCE center

Go to this title's Internet Resource Center and read the article titled "The National Electronic Disease Surveillance System (NEDSS)." www.emcp.net/CUT4e

The Importance of Accurate Data

Databases are records of events or situations, so they must be continually updated to ensure that the data they contain is accurate. Consider, for example, the situation of an insurance company that maintains a database containing names, addresses, birth dates, and policy information. Several different departments in the company may share the database. Marketing sends out information whenever policy benefits change. They also send mailings to people who may be interested in new benefits. The accounting department uses the database for mailing billing

Paper forms served as a primitive type of database before records were computerized.

Inventory data and sales data must be accurate in order to manage a business.

notices. Company managers regularly generate reports, or information combined from linked database tables, to forecast new directions for the firm. If a policyholder moves to a new address, the database must be updated to reflect the change, or the information the company uses will be incorrect. Any inaccuracy will create an avalanche of problems throughout the different company departments and will reduce the quality of the data in the database.

Data quality is particularly important when considering that summarized information is often used for decision making. Every retail clerk, for example, helps tally store sales. Each entry is stored in a database that constantly tracks a store's current status. From the company's viewpoint this task is of vital importance. Sales information entered into the database allows managers to decide what supplies need to be ordered, and it provides a constantly updated snapshot of revenue.

Levels of Data within a Database

Tech Demo 9-1
Database Structure

The ability to organize and reorganize data for different purposes is due to two database characteristics: their vast storage potential and the way they organize data. Traditional databases organize data in a hierarchical fashion, containing information about entities in the form of fields, records, and files. Figure 9-1 illustrates this hierarchy.

Entities An **entity** is a person, place, thing, or event. Database files record information about different entities using fields, records, and files. A typical entity might be a sales transaction that describes products removed from inventory and the amount of money received for those products. Other examples of entities include student grades, traffic violations, and telephone records.

Fields The smallest element of data in a database is a field. A **field** is a single value, such as a name, address, or dollar amount. A field generally has three attributes:

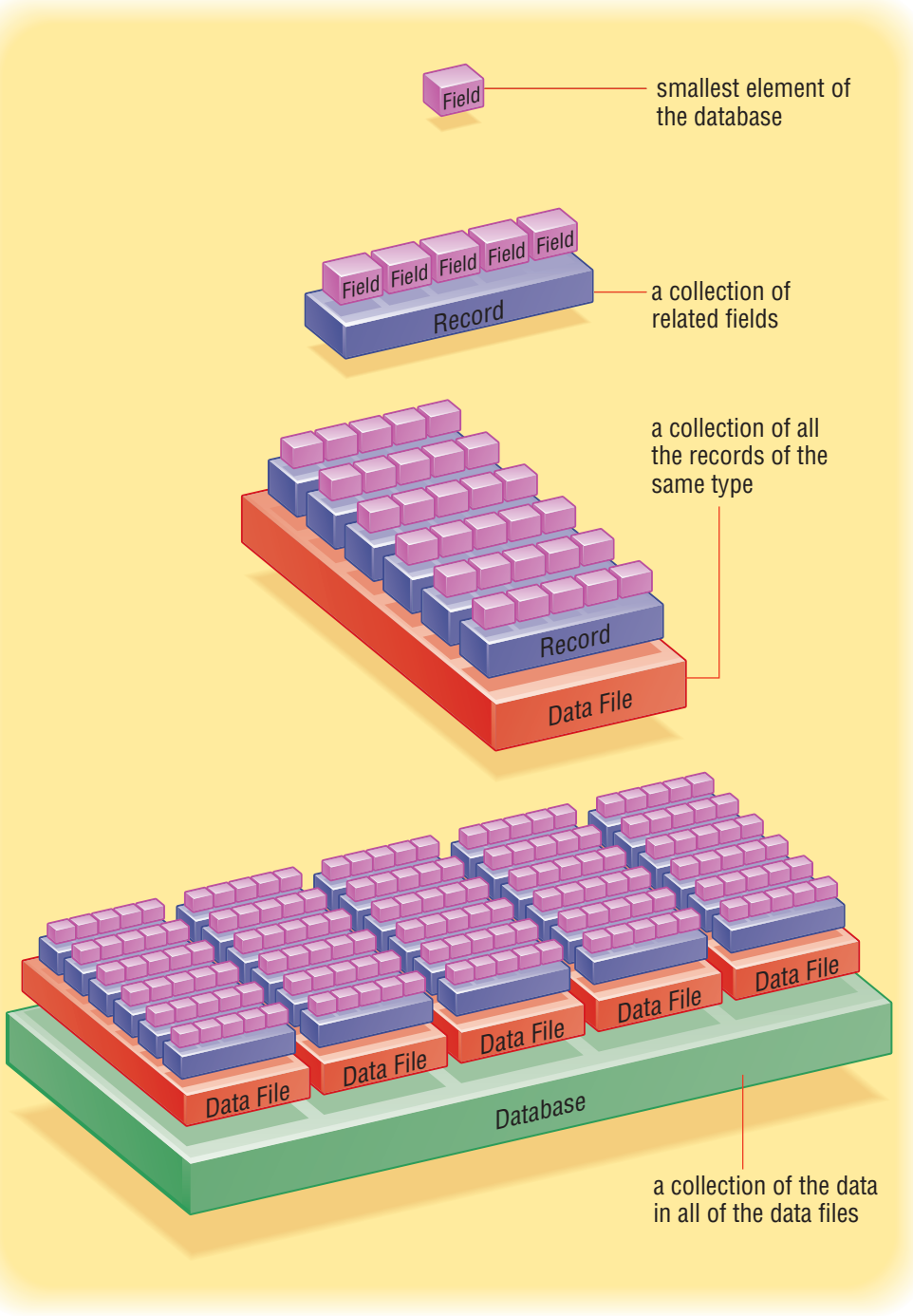


Figure 9-1 Traditional Database Hierarchy
A traditional database, as shown above, organizes data in a hierarchy. The field is the smallest element. Fields are located within a record, and records are organized in a data file. The database is made up of a collection of data files.

- data type: usually numeric or text (numbers and letters)
- name: assigned by the person developing the database
- size: the number of characters that can be entered

A field called *FirstName* might have a size of 20 characters. The database automatically would truncate any name longer than 20 characters to fit into the field.

Medical Records Go HITECH

IMAGINE YOU'VE JUST MOVED to a new city, and you need to see a doctor. What if your new doctor could access all of your medical records without waiting to receive them from your previous doctor's office? What if he or she could gain a thorough understanding of your health history instantly, just by clicking a mouse? That's exactly what the government thinks doctors should be able to do, and why it is helping to fund the creation of a massive new networked database—one that would contain the electronic health records of every American.

On February 17, 2009, President Obama signed the Health Information Technology for Economic and Clinical Health (HITECH) Act, which provides federal stimulus money to healthcare providers who adopt an electronic system for keeping patient medical records, and will likely carry penalties for those who do not do so by 2014. To receive funding, the provider's system must be interoperable, meaning the records can be viewed and shared by other providers. How exactly the records would be shared is still up in the air. While there has been talk of a national network, in which all the providers would have to agree on a standardized method of presenting information, some worry that this would create more problems than it solves. Instead, they envision a "network of networks," which would be similar to the system used to allow customers to access their bank accounts from any ATM across the world.

Because less than 10 percent of U.S. healthcare providers currently use electronic records, the task of creating this networked database is daunting and will take years to complete. However, the government believes that in the end, the system will save money, boost the economy, and most importantly, help Americans live healthier lives.

The most common data types are numeric and alphanumeric. **Numeric data** consists of numbers only. **Alphanumeric data** consists of letters, numbers, and sometimes special characters. Other types exist, such as logical yes/no fields, time and date fields, and memo fields, which allow unlimited character input. The more unusual data types include pictures, movies, and sound. Table 9-1 describes various field data types.

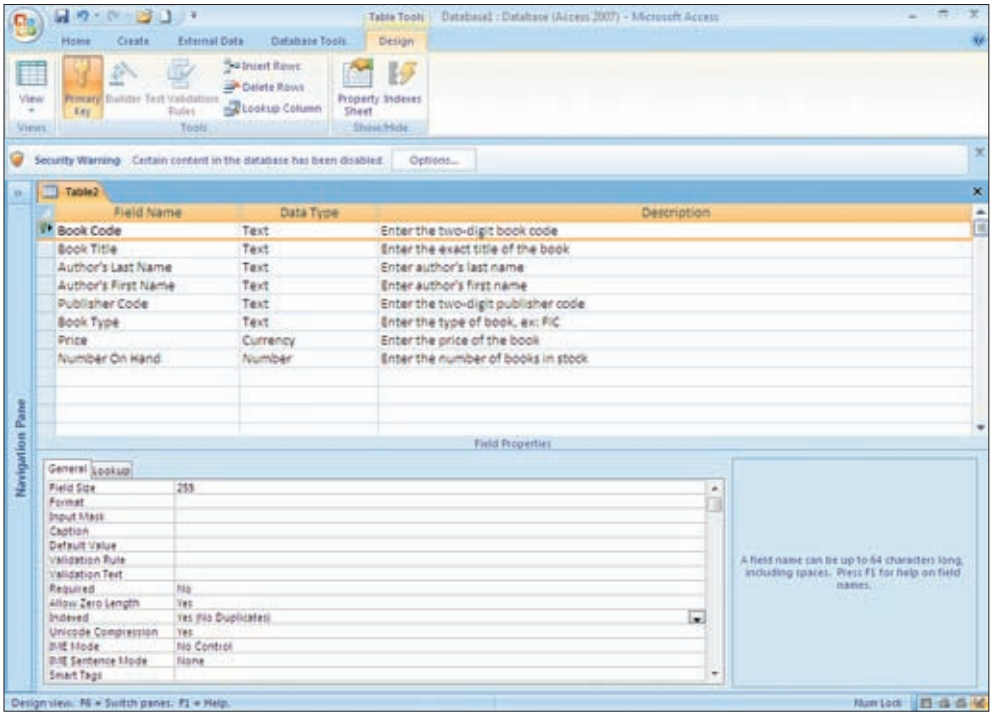
Records A collection of related fields describing an event or situation is called a **record**. If a record covers mailing information, it would likely include fields for *name, address*, *city*, *state*, and *postal code*.

Files A **database file** is a collection of records of the same type. When a database is designed and built, the designer must decide what records will be used, which fields will be in those records, and which data type and size each field will have. The record layout is used as the basis for each record in the table. For

Table 9-1 Common Field Data Types

Data Type	Description of Data
alphanumeric	textual information such as a person's name
numeric	numbers such as a count of inventoried products
logical	yes/no states such as "married" or "retired"
currency	dollar amounts such as a bank account balance
memo	lengthy text information, notes, or history
object	nontextual information such as pictures
hyperlink	connects a record to a Web address
date/time	allows only valid dates/times, such as November 12, 2007
sequence (or autonumber)	integer that automatically increases whenever used; generally is used to make a primary key that automatically changes in order to be unique

example, if a company's database has a record type called CUSTOMER, then there would be a record of CUSTOMER TYPE for each customer. All the records would use the same set of fields. When all the records of the same type are combined, they become a database file.



Deciding what fields to include in a record is part of database design. This screen is the design screen in Microsoft Access 2007.

Databases and Information Systems

Networked databases allow businesses to save time and money by coordinating their operations. If each department in a business kept its own customer records, there would be duplicate entries, wasting time and causing confusion. If the different departments share a single networked database, information has to be entered only once, and it can then be accessed freely by anyone needing it.

Tech Demo 9-2
Web Databases

The most common database application is an **information system**, which is a system of computer hardware, software, and operating procedures. Information systems are designed to meet the information needs of a company, such as tracking income and expenditures, recording transactions, sharing resources, and planning for future growth. A variety of information systems has evolved as software capabilities and market needs have changed over the years. This chapter provides a brief overview of information systems as they relate to databases.

Management Information Systems

A **management information system (MIS)** is used to track and control every transaction through a database. A **transaction** is a business activity central to the nature of an enterprise. A transaction can be the sale of a product, the flight of an airliner, or the recording of a college course grade. A database stores the information that is at the core of any MIS system.

An MIS for an airline flight information display would need to handle flight numbers, gate numbers, and expected arrival and departure times. This information is available to the public and company employees to view. The data comes directly from a computer running an MIS. Every attendant at the ticket counter who enters data using a keyboard is communicating with the MIS. When a flight

An airline's MIS would contain information about passengers, luggage, seat assignments, and flight arrival/departure times.

Super-Secure Soccer

THE GERMAN SPORTS ESTABLISHMENT was proud and excited to host the 2006 World Cup Soccer Tournament. Yet with honor came the responsibility of protecting the large crowds gathered to watch the matches. Certainly no one wanted a repeat of the 1972 Munich Olympics, at which a group of Palestinian nationalists raided the Olympic village and killed 11 Israeli athletes.

But when the 64 games of the World Cup were over, no significant incidents had occurred—due no doubt in large part to the unprecedented security measures and the widespread use of databases to share information among agencies, venues, and security sites.

Operating 24 hours a day during the weeks in which the games took place, the National Information and Cooperation Center at the German Interior Ministry in Berlin monitored scores of computers fed from surveillance cameras in and around stadiums. Some of the cameras were equipped with biometric readers that could check the facial features of suspicious persons against a database of known soccer troublemakers. Thousands of tips from authorities in participating countries also contributed to the hooligan database. The computers also gathered and shared reports from police, Interpol, and intelligence services.

At the arenas, some of the 30,000 police officers on duty were equipped with wireless fingerprint devices that allowed them to instantly submit an optical scan of fingerprints taken from detained suspects into a central database at the German Federal Intelligence Service.

The 3.5 million ticket buyers had to provide name, address, date of birth, nationality, and ID card or passport number—more personal information than has ever been asked of a soccer fan. As fans entered the stadium, ticket takers scanned the RFID chips embedded in the tickets to look for matches in the database. The goal was to prevent entry to hooligans and known criminals, and also to thwart ticket scalping.

Germany took many other safety measures as well. Thousands of troops were on alert. Fifteen thousand private security specialists were hired for the games. Airspace over all the matches was closed. But the use of database and wireless devices were the hands-on technologies that helped deliver a safe and secure soccer scene for the 2006 World Cup in Germany.

finally does depart, another MIS run by the Federal Aviation Administration (FAA) uses radar systems to track its progress toward its destination.

Office Information Systems

First popularized in the 1960s, the concept of an **office information system (OIS)** was billed as a replacement for paper-based information systems. An **electronic office** is an office that implements an OIS. Many people thought that the advent of electronic offices would lead to "paperless offices." Unfortunately, computer systems tend to generate more paper documents than their noncomputerized counter-

parts. This is in large part due to the ease with which printers can now churn out documents, as compared with the precomputer era when documents would have been typed manually.

The electronic office today is the norm rather than the exception. Most organizations of any size use a computerized OIS to manage their operations. For example, a shipping company OIS may hold records for every customer and corporate account and may use them to identify key clients. When an important client calls in, the receptionist can look him up in the database and immediately direct his call to the right agent. Clerks can also quickly access company work orders and check the status of any client orders.

Decision Support Systems

A **decision support system (DSS)** is another common form of information system. Rather than simply tracking the day-to-day operations of a business, a DSS is designed to help management make decisions about an operation. A DSS might include a predictive model of the business that allows managers to work with "what-if" scenarios. If a business is considering expansion or the release of a new product, a DSS can help determine if the change would be likely to succeed or fail. The database could provide information on past performance, which the business owners could use to judge the cost of the expansion and any changes to revenue that might result.

Factory Automation Systems

Computer-aided manufacturing (CAM) and **computer-integrated manufacturing (CIM)** are information systems that support factory automation. Generally, CAM refers to systems that run an assembly line directly, controlling the manufacturing process from the shop-floor level of conveyor belts and robots. CAM systems form a portion of a complete CIM system, a higher-level concept indicating a

CIM and CAM systems are used to automate manufacturing.

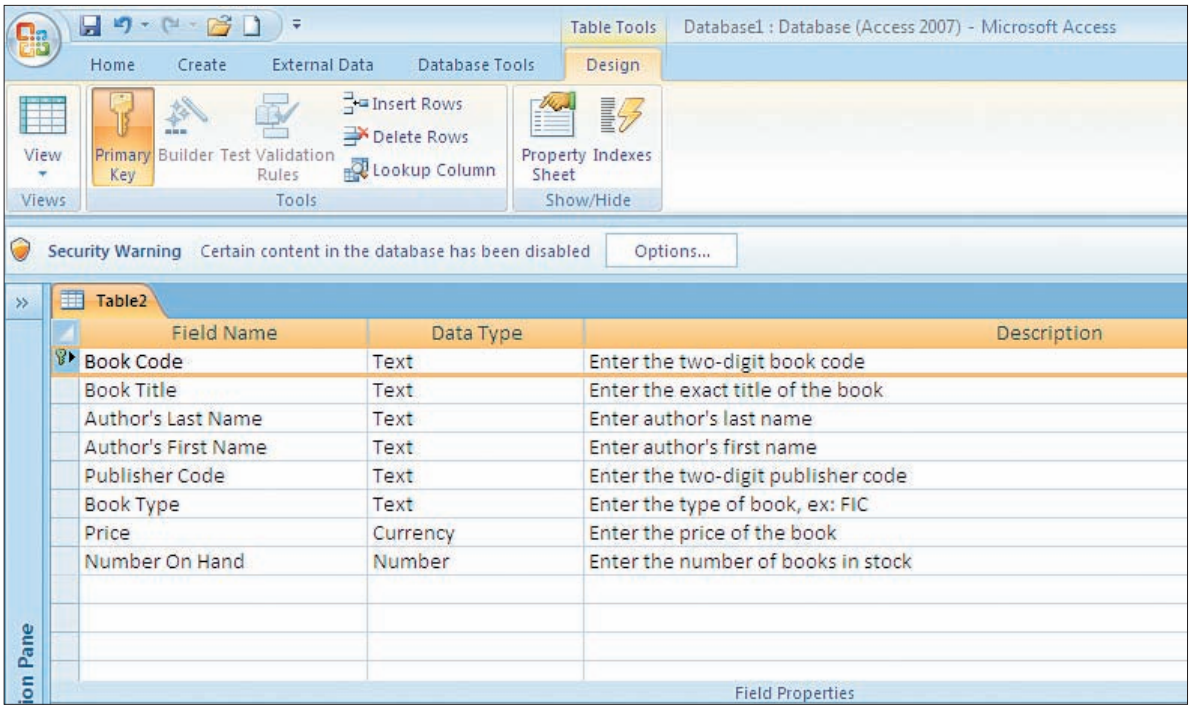
system that controls a manufacturing process from beginning to end. The database at the core of these systems stores information about factory operations, including counters that automatically add each item manufactured to inventory as it is made.

Database Management System Software

Database management system (DBMS) software is software that allows a user to create and manage a computerized database. A DBMS also allows a user to create reports from stored data. IBM's DB2,, Oracle from Oracle Corporation, and Microsoft's SQL Server dominate the enterprise DBMS market. In the PC market, Microsoft Access is a widely used DBMS. In addition to maintaining the overall structure of the data, a DBMS has many other functions. Database keys, query tools, security elements, metadata, and backup and recovery utilities allow users to manipulate database data into information.

Database Keys

A **key** is an attribute that can be used to identify a set of information and therefore provide a means to search a database. Within a database, fields are used as keys, and the designer designates the most important field in a record as the primary key. The **primary key** must also be unique, so it can be used to locate a record quickly. First names are poor keys because many people have the same first name. In a database that tracks traffic and parking violations, the state of registration plus car license plate number could be used as a primary key because no two cars registered in the same state can have the same license number. Phone, bank



In Microsoft Access, the Primary Key button allows the user to designate the table field that will function as a primary key. To activate the feature, users select the desired table field and then click the Primary Key button on the Table Design toolbar.

CEO, ORACLE CORPORATION

Larry Ellison

LARRY ELLISON, COFOUNDER AND CEO of Oracle Corporation, is one of the most flamboyant and outspoken entrepreneurs of this century. Ellison is one of the richest as well. With a net worth of $50 billion and a financially successful company, he has reached the pinnacle of corporate success.

Among some of his friends and peers, Ellison may be considered an eccentric. It has been rumored that he once tried to purchase a Russian MIG fighter plane, but U.S. customs would not allow the plane into the country. In another incident, Ellison reportedly upset San Jose airport officials by landing his private jet after an 11 p.m. curfew, an action resulting in a $10,000 fine. To many of his fellow business associates, however, he is a business genius, a marketing whiz, and an avid promoter of simplified computing machines and network computers.

Oracle is the world's leading supplier of information management software, and the world's second-largest independent software company, boasting annual revenues of more than $9.7 billion. The company's diverse product line includes database software products, applications servers, software information management tools, and software suites for electronic commerce applications. Oracle's database software is the most widely used corporate database worldwide, and the company's product line continues to expand at a rapid pace.

account, credit card, social security, reservation confirmation, and work order numbers are commonly used as primary keys because they are unique. Primary keys are usually large numbers, but they may also be alphanumeric, like many drivers' license numbers.

It is possible to search a database without using the primary key, but it will be more difficult as other keys may not be unique and therefore may return a confusing array of results. This is why customers are usually asked to provide some form of unique identification when they call a company to inquire about a billing error or a product order. The requested information acts as a primary key, enabling the person assisting them to locate records quickly. If a service representative asked for a name, he would have to sort through records of everyone with the same name. This would waste time, as there could be thousands of duplicate names in a large database. Managing the enormous amounts of information in a database would be nearly impossible without primary keys.

Users may need to browse records to find the desired information when a record's primary key is unknown. **Data browsing** is the process of moving through a database file, examining each record, and looking for information. This strategy is also used when a search on a secondary key returns many results that must be sifted. Hunting through the best results returned by a search engine is a form of data browsing.

Query Tools

Databases are stored in the form of data files until the system needs to perform **file processing** on the information. At that point, it is copied into RAM. Databases are sometimes too large to fit into RAM all at once, so only portions of the total data can be manipulated at any one time. To work with large amounts of data, database management systems come equipped with **query tools** that help users narrow down the amount of information that needs to be searched. A **query** allows users to ask questions designed to retrieve needed information. For example, a query combined with a report can be used to ask a grades database to list all students in the top 10 percent of academic achievement. The results could be used to print a report that would be the dean's list for that semester.

Requesting information involves the use of a query language. **Structured Query Language (SQL)** is the most popular database query language. It is simple when compared with a programming language, but it is also "structured," meaning that it is not as freeform as natural programming languages that mimic human speech. SQL is a very commonly supported query format and works with most databases in use today. Since it is standardized and so widely supported, SQL is often used to bridge communications gaps between database systems running on computers employing different operating systems.

The basic query command that SQL commands is the **SELECT command**, which asks a database to return records that match specified criteria. The command uses the keywords SELECT [fields] FROM [table] and specifies the table and fields from which the information is to be selected (see Figure 9-2).

Information from one query can be used to pull up data from more than one record source through a process called **joining**, which matches data from fields in various database files. This is usually done by matching up primary keys used as fields in one record to specify a relationship to another record. For example, if student names and grades were entered as separate records, "out of state" could be used as the primary key to find the grades of out-of-state students (see Figure 9-3).

Security Measures

A DBMS also provides **security measures** to protect and safeguard data. Payroll, accounts receivable, and e-mail storage systems all contain sensitive information

```
SELECT     Student.FName, Student.LName, Student.StuID
FROM       StudentData
WHERE      Student.GPA>3.49
AND        Student.Enrolled="YES"
```

Figure 9-2 Example of Structured Query Language
This SQL statement could be used to search for the top students in a college, with "top" defined as those students with a grade point average (GPA) above 3.49.

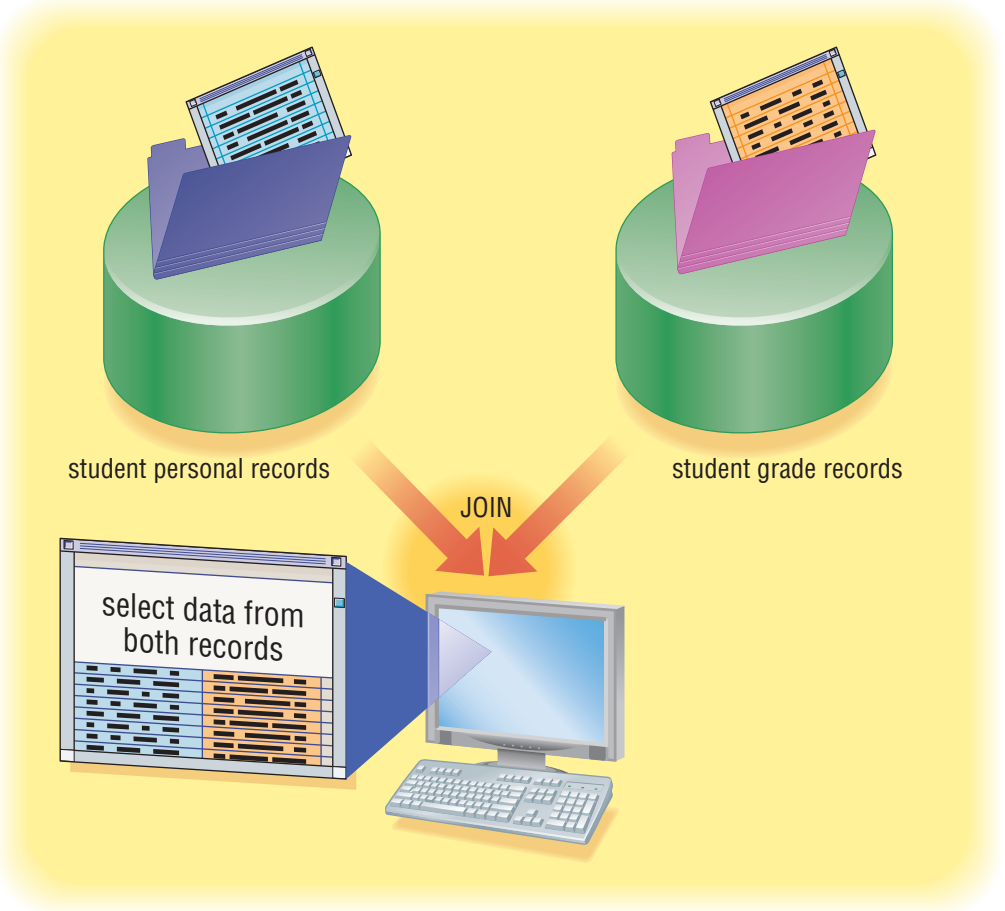


Figure 9-3 Joining
Joining matches data from fields in various database files. Both the student personal records and student grade records files would have a common student ID number field, allowing the information from both of these files to be returned in a single query.

that must be protected against theft, alteration, or deletion. Competitors, hackers, crackers, or disgruntled employees can do a great deal of harm if they are allowed access to critical company databases.

Maintaining data security is a critical issue for database administrators. Security methods include restricting access by requiring user identification and passwords. Usually only users with higher access levels can change data in a database, and only those with the highest level of access can change the format of the data itself.

Metadata and the Data Dictionary

Metadata is information about data. **Data dictionary** is the term for a body of metadata. Metadata can be used for many things, but often describes the significance of various elements of a database. If a record field was called FNAME, metadata might describe this as being a text field with a maximum length of 20 characters, used for holding a person's first name. These characteristics might not be immediately obvious from the name FNAME. Designers, users, and administrators use metadata to manage databases.

Legacy Database Access

A **legacy database** is a database that runs using languages, platforms, or models that are no longer supported by an organization's current database system. In order to be able to continue to access the information stored on these databases, their programming code must be made compatible with the newer system. An effective way to do this is to migrate legacy applications to operating systems that follow open or standard programming interfaces, which will allow a company to use its legacy applications on most operating systems.

Another option for accessing legacy databases is to use software tools that allow the data to be read directly by newer database systems. This avoids the need to convert the existing code to newer code, saving time and money. These tools are often integrated into database management system software.

Backup and Recovery Utilities

Another major element that DBMS systems provide is a method for backing up and recovering lost data. Almost all companies keep sales, marketing, customer, payroll, and tax records on a database. To lose this data would be disastrous, so regular backups are made and stored safely. If a **recovery**, or restoration of data, is needed, the most recent backup is used. Important backup information is often stored in safes or off-site.

Types of Databases

Databases are often categorized by the way they organize data (data models), or by their function (storing information for ongoing processes versus storing information for historical purposes).

Databases Classified by Data Model

A **data model** defines the structure of information to be contained in a database, how the database will use the information, and how the different items in the database relate to each other. The goal of a database model is to identify and organize the database contents in a logical way.

The data model employed by a database is so central to the way it works that most databases are named after them. The data model chosen matters primarily to the database developer, as most data models can provide any kind of data or interface. Advanced data models tend to be more reliable and consistent, allowing for greater connectivity with outside systems. They also tend to be easier to work with and less expensive to develop and maintain. Common data models include flat file, relational, object-oriented, multimedia, and hybrids.

Flat File Databases A traditional data file storage system that lacks the ability to interrelate data in an organizational structure is known as a **flat file database**—flat because the database contains only one table or file. These systems are simple and easy to use, but in general flat file systems are slower to respond because the records must be searched sequentially. They also consume more disk space than other types of databases. Flat files can be useful for finding information, but relational database systems offer faster, more comprehensive reporting capabilities.

	ID	User Name	Password	Last Name	First Name
▶	1	mingdas	*******	Solen	Mingda
	2	kimmerr	*******	Merrifield	Kim
	3	klamoure	*******	LaMoure	Keith
	4	rosemh	*******	Hettinger	Rose
	5	gregjw	*******	Winston	Greg
	6	franklee	*******	Lee	Frank
	7	hwilson	*******	Wilson	Hannah
*	(New)				

In a flat file database, a file is a collection of records. Each record is contained in a separate line, and fields are separated by a delimiter such as a comma.

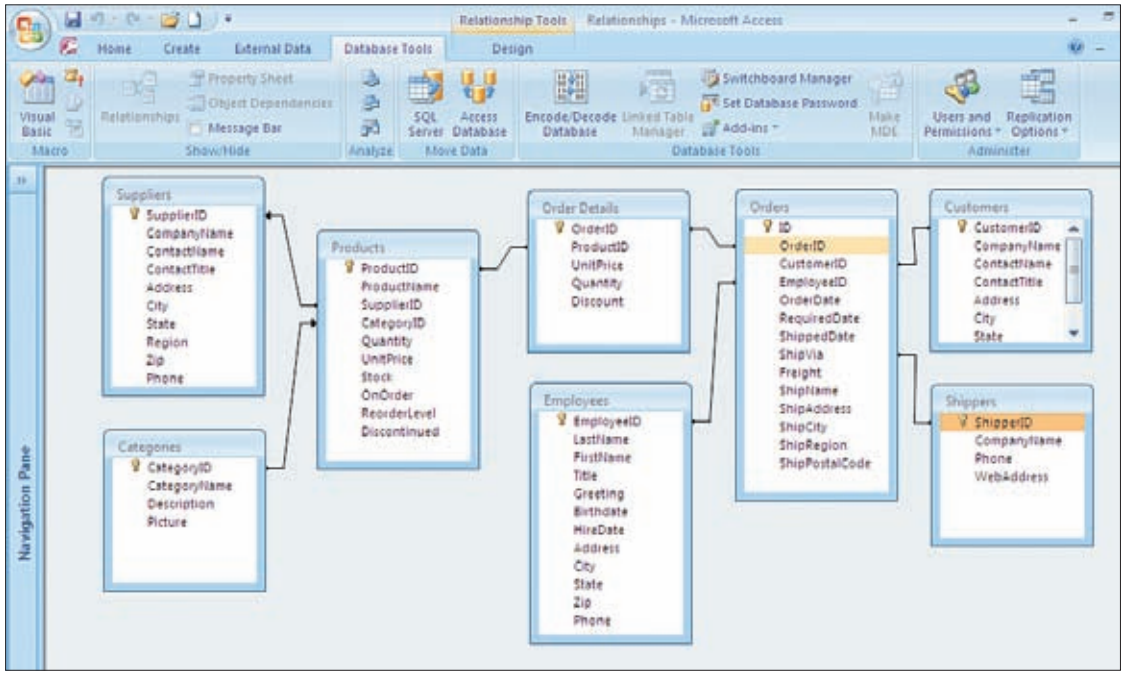
Relational Databases Most modern databases use a **relational database** model in which fields can be shared among all the files in the database, making it possible to connect them. In a relational database, a file is called a **table** (consisting of rows and columns), the record is called a **tuple**, and a field is called an **attribute**. Although many of the element definitions are the same as with a traditional flat file system, the primary difference concerns the organization of the tuples and their relationship to the table they are in.

Compared with flat file models, the most important advantage relational database models have is that information can be shared, extracted, and combined more easily among different tables. In a traditional system, an entity's address might be stored in several different records. If the entity's address information ever changes, address records would have to be updated in every location. If any record was overlooked, it would contain outdated information, meaning that any output using that record would be inaccurate. In relational databases, rather than having an attribute (field) called *address* repeated in many records, it would be stored once and shared among multiple tables. In a relational database, a single change to an attribute would change that attribute in all the tables at once.

Object-Oriented Databases An **object-oriented database** stores data in the form of objects (units of object-oriented programming logic). Each object contains both the data related to the object (such as the fields of a record) and the actions that the user might want to perform on that object. A record object created for a payroll department might contain the salary of an individual, along with the calculations required to withhold taxes and print out a paycheck. Object-oriented databases allow for faster development and access times, speeding up database functions and making their development easier.

Go to this title's Internet Resource Center and read the article titled "Relational Database Design."
www.emcp.net/CUT4e

Tech Demo 9-3
Relational Database



Microsoft Access shows the relationships between tables in a database. Attributes such as the customer number, customer's street address, and customer's phone number are stored in the Customers table while the customer number would also be found in the Invoice table when a purchase has been made.

Multimedia Databases As computer storage and processing speeds continue to increase, so do the number of multimedia databases. In addition to the text and numbers a typical database model handles, a **multimedia database** allows the storage of pictures, movies, sounds, and hyperlinked fields. A multimedia employee database file, for example, could include snapshots of employees, along with recordings of their voices.

Hybrid Databases A database is not limited to employing a single data model. Several different models may be used to allow more effective data handling. This type of database is called a **hybrid database**. A combination of relational and object-oriented database models is a popular form of hybrid database. Relational databases can handle only simple mathematical calculations, such as addition or subtraction, so users may have to use separate applications to perform complex functions. Combining relational and object-oriented data models allows for more sophisticated analysis within the framework of the relational database. The advent of the Web and the increasing availability of multimedia files is another factor increasing the popularity of this type of hybrid.

Databases Classified by Function

The two major functional classifications for databases are operational databases and data warehouses. Operational databases are working systems that are continually being updated. For example, a supermarket may use an operational database that is updated every time a shopper purchases an item. In contrast, a data warehouse is a system from which data is extracted and analyzed by a company's staff

An information system used in a prison would combine alphanumeric prisoner data with a mug shot image in a hybrid database.

in order to make strategic decisions. A data warehouse typically is not modified or updated with new information.

It is possible for a database system to perform more than one function at the same time. Also, these classifications are not generally associated with a specific data model. Operational databases, for example, might use flat-file or relational structures. The list of database types is still evolving, and new classes may emerge in the future. Web databases are one example of a function classification that has evolved in recent years.

Operational Databases An **operational database** works by offering a snapshot of a fluid situation. These systems are called operational databases because they are usually used to track an operation or situation, such as the inventory of a store. Examples of operational databases include inventory tracking systems, such as that used at a large retail store. The store has products in stock and on the shelves for customers to purchase. As cashiers scan the bar codes on merchandise, the transactions automatically update a database that tracks the amount and location of each item in the store. When the stock of diapers or lawn chairs runs low, the database automatically orders more.

The operational database at a nursery is updated each time an item is scanned at a checkout station.

E-commerce Web sites are based on operational databases. These sites allow users to place items in a virtual shopping cart and enter credit card and shipping information without ever talking to a person. This data is entered into a database, which is later used to track orders and payments.

Biopirates Beware

INDIA IS SEEKING TO PUT THE PRESSURE ON BIOPIRACY, the practice of poaching the knowledge and resources of indigenous peoples. To protect its natural and cultural property, the country's National Institute of Science Communication and Information Resources has compiled a database of more than 100,000 traditional herbal medicines and thousands of plants and yoga positions. The database also contains millions of pages of ancient Indian text, national recipes, and Indian farming and architectural methods.

India has already turned back some biopirates. U.S. and European players tried to patent the medicinal properties of the spice turmeric and the Asian evergreen neem, but India won the legal battles to overturn the patents after proving that the medical properties were already well known in that country.

To combat further piracy, India and several other developing nations are urging the World Trade Organization to establish a system for controlling how native animals, plants, and lore are used to make pharmaceuticals or other products. The WTO wants laws that would require patent applicants to state the country of origin of any plant or animal used, to seek permissions from that country, and to share revenue from any resulting product.

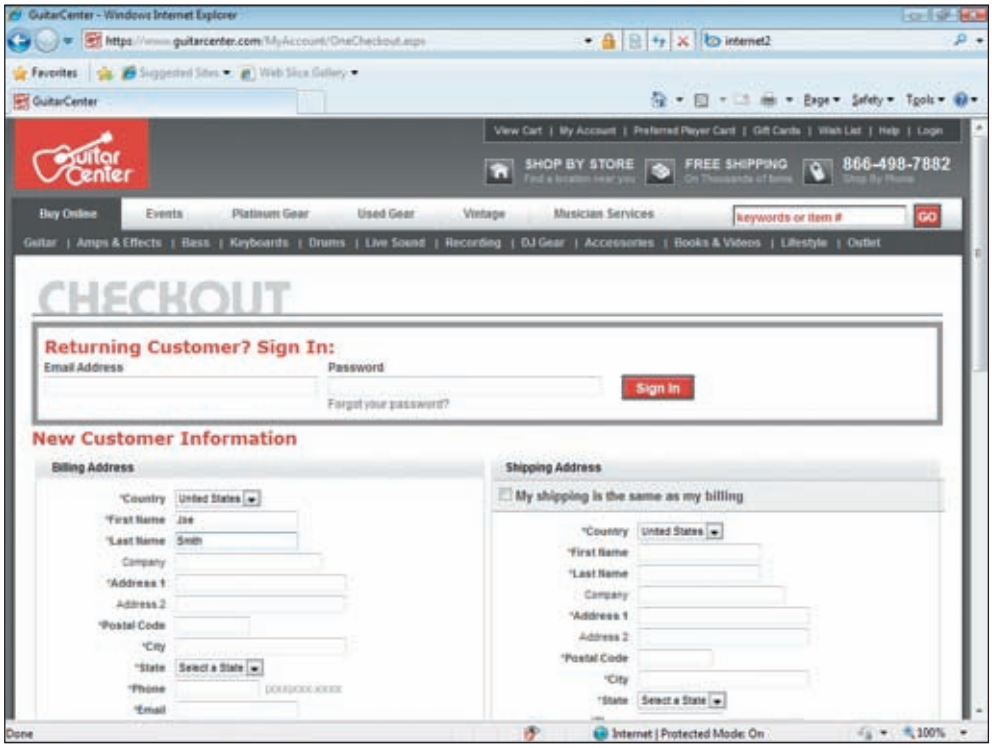
Putting together a system to protect national property won't be easy. It would be an enormous task to track down the origins of particular ideas or biological organisms. Knowledge and tradition that have been passed down orally would be difficult to track to their roots. The development of a system that would assign property rights and establish ownership would require a Solomon-like ability to determine heritage. Nonetheless, India hopes that by establishing a database of national property, it has ammunition to fight outsiders who want to claim yoga or Darjeeling tea as their own.

Depending on the amount of traffic that they receive, Web site databases may be distributed databases. A **distributed database** is spread across multiple networked computers, with each computer storing a portion of the total amount of data. These databases are valued for their ability to hold more information than any one computer can contain, and they are cheaply and easily expanded. Distributed databases offer advantages in cost, expandability, and storage capability. A major disadvantage is that they do not operate as quickly as mainframe or supercomputer databases, where all information is contained on one computer.

INTERNET RESOURCE center

Go to this title's Internet Resource Center and read the article titled "Data Warehousing and Data Mining." www.emcp.net/CUT4e

Data Warehouses A **data warehouse** is used to store data gathered from one or more databases. Unlike operational databases, data warehouses do not change, delete, or manipulate the information they store. As their name implies, data warehouses function as vast storage places for holding information that can later be used in a variety of ways.



E-commerce Web sites use operational databases to store purchase information and customer information.

Law enforcement records are one example of a data warehouse, with vast collections of photographs and arrest records on file for reference or for statistical use. Companies also maintain data warehouses to test new ideas against past results. Many of the largest company databases contain information dating back to the 1950s. Companies are increasingly experimenting with these databases, using data mining techniques to sift through information and identify previously unnoticed trends. The strategy resembles survey work, but produces real results rather than estimating future behavior. Data mining systems often include elements of artificial intelligence.

Data warehouses allow customer service representatives to suggest new products to customers based on their demographics and past behavior.

Planning and Designing Database Systems

Planning and designing a database system requires a combination of knowledge, skills, and creativity. This job is usually handled by a **systems analyst**. A systems analyst may also become responsible for administering the database after it is built. Like a structural engineer who designs buildings and draws blueprints, a systems analyst is an experienced specialist who can identify project needs and then design a structure or system to fulfill them.

The Database Management Approach

The development and maintenance of database structures and applications employs a methodology called the **database management approach**, sometimes shortened to the database approach. Using the software tools of a database management system (DBMS), a systems analyst or database designer follows three broad steps in the planning process:

Tech Demo 9-4
Database Design

1. **Create an organizational structure for the data.** A designer first focuses on creating an abstract structure that imitates a real-life situation, drawing on an analysis of the information needs and the purpose(s) of the database. Decisions are made as to the kinds of records needed, the number of fields needed in each record, and how fields should be associated to form records. Choosing the data type and formatting for each field is another basic decision. With the field-record-file structure established, the database is ready to accept data.
2. **Design an interface that makes the database user-friendly.** Users do not need to understand a database's internal structure. Their most important consideration is that they are able to easily enter data and request information. A **front-end interface program** makes this possible. It interacts with the DBMS, the software that manipulates and manages the database. One of the primary functions of the front-end interface is to prevent erroneous data from entering the database. Figure 9-4 shows the relationship of the DBMS, the front-end interface, and the database.
3. **Set up reporting capabilities to allow for inquiry and response.** Because these functions represent the main database information analysis tools, they are a major focus of the designer's work. A DBMS typically includes reporting features, and the designer chooses how and where in the system the features can be used.

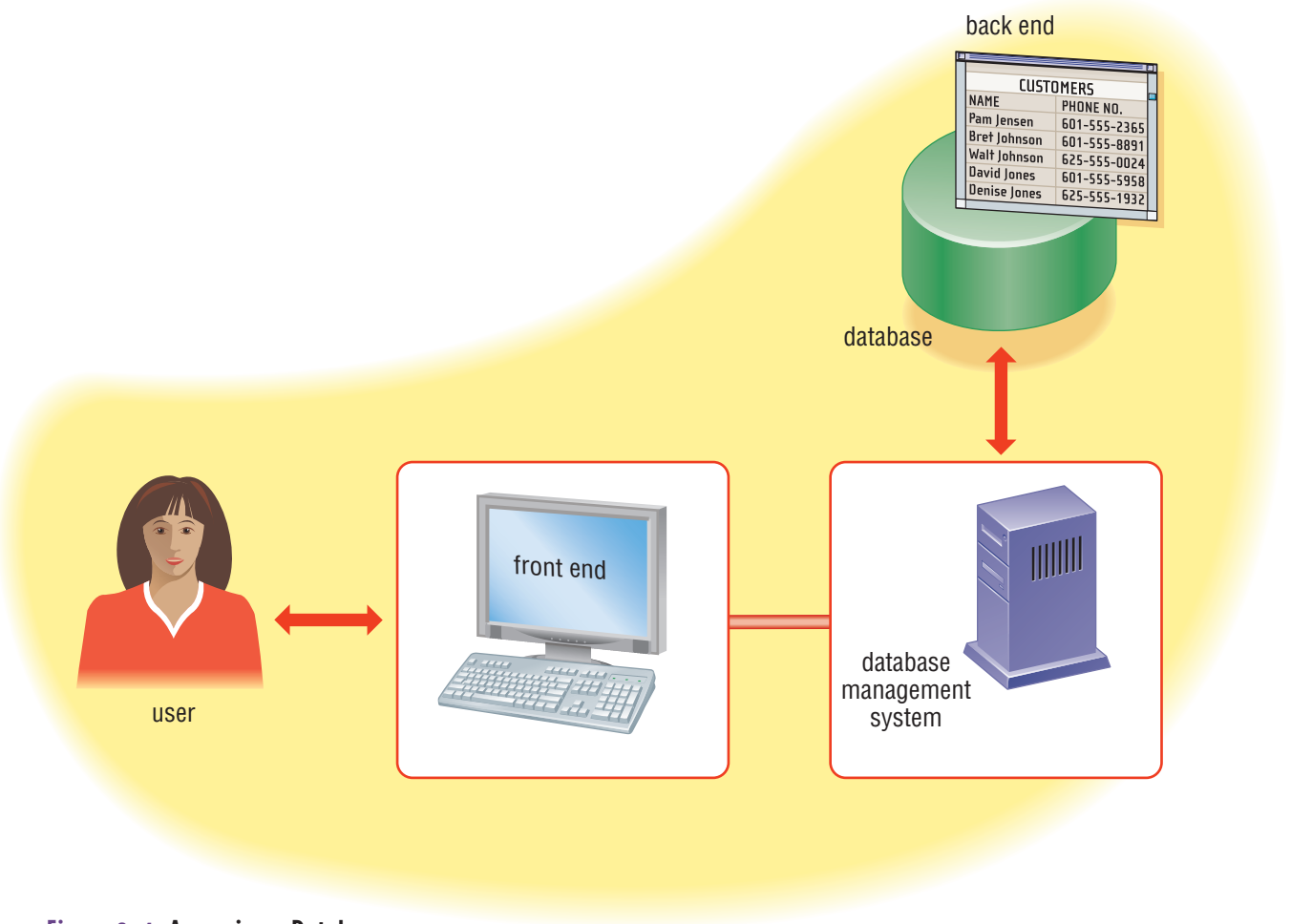


Figure 9-4 Accessing a Database
The user requests and enters data through the front-end interface program. The database management system manages the retrieval and update of the database itself.

Database Objects: Tools in the DBMS

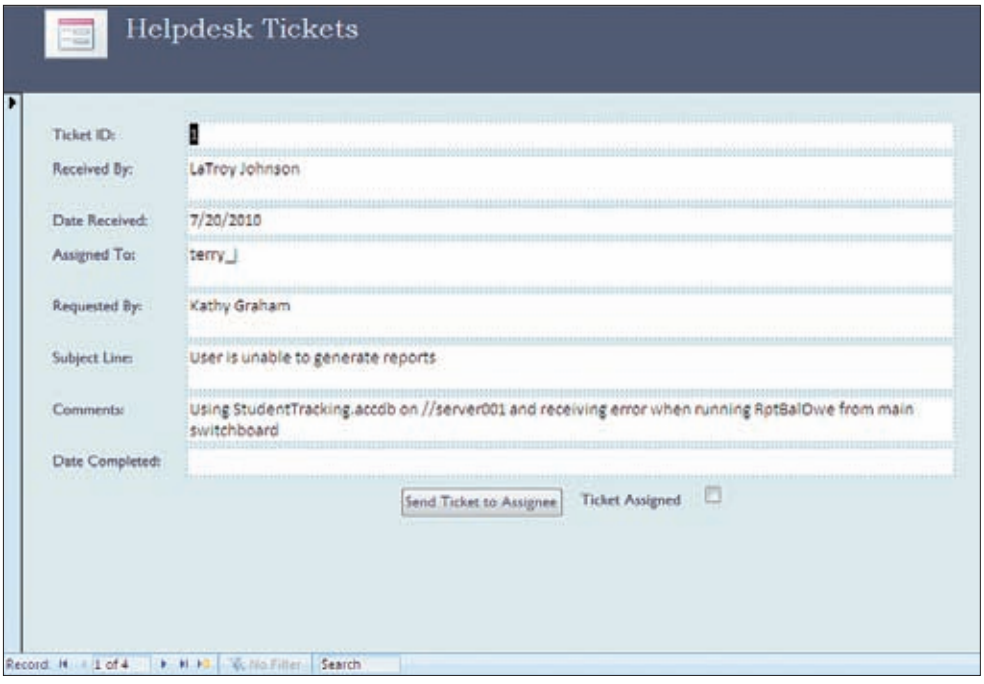
A **database object** is an element within an object-oriented database, such as a form, a report, or a data filter. Database designers use database objects to build the system interface and the reporting features.

Forms A **form** is a template that allows users to enter data into the database. Forms perform the important jobs of detecting and preventing erroneous or incomplete data input. A form can be configured to allow the input and display of data in any fashion that the system designer sees fit. Forms can allow users to search through data, move to other forms, print reports, and make changes. Many jobs involve working with forms in a database, such as airline ticketing and hotel front desk work.

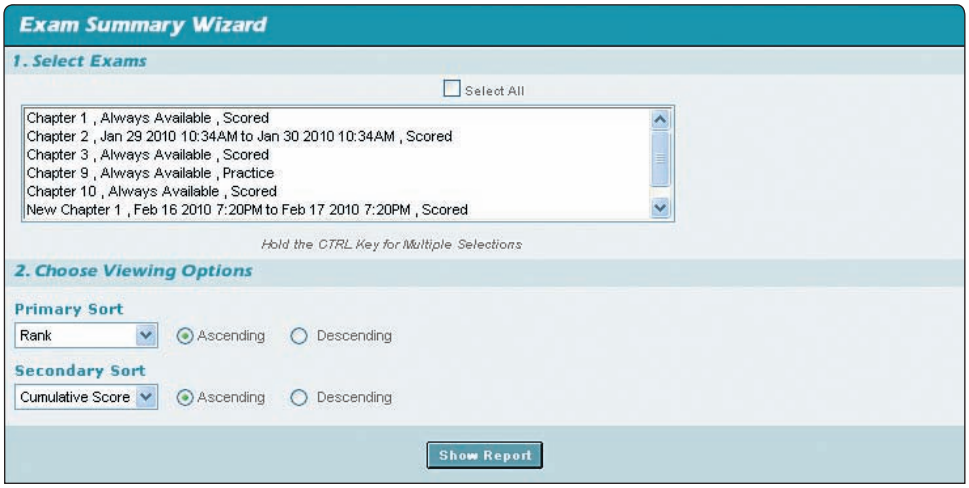
Forms are designed to let users input their answers to the form's questions or prompts. Default answers are included to ensure that the database has all the data required for its fields. Defaults are assumed input that will be utilized in the absence of entered input. Many software programs feature dialog boxes containing questions with default answers. If a user simply presses the *Enter* key, or clicks OK, the program will assume the default input for that question. For example, if a program asks "Do you wish to save before exiting?" and a user presses the *Enter* key, it will assume that the work should be saved, which is the default option.

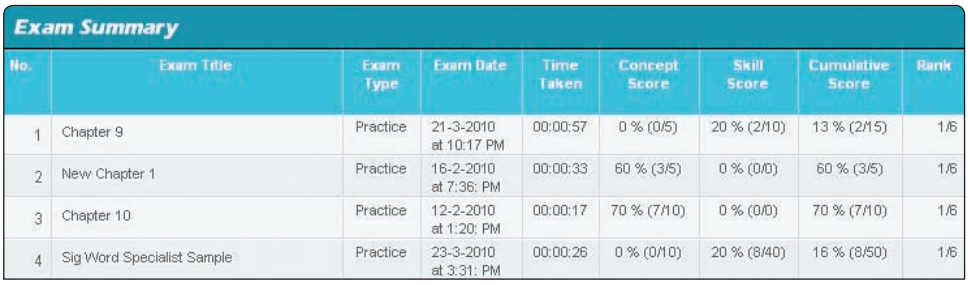
Reports A **report** is a formatted body of output from a database. Most reports are designed to be printed out for later review. Monthly phone bills, report cards, and grade transcripts are all examples of database reports.

A report can be generated by request or automatically. For example, credit checks are a type of database report that is not issued on a regular basis. Instead,



A form is an easy way to input data into a database table.


Exam Summary Wizard
1. Select Exams
Select All
Chapter 1 , Always Available , Scored
Chapter 2 , Jan 29 2010 10:34AM to Jan 30 2010 10:34AM , Scored
Chapter 3 , Always Available , Scored
Chapter 9 , Always Available , Practice
Chapter 10 , Always Available , Scored
New Chapter 1 , Feb 16 2010 7:20PM to Feb 17 2010 7:20PM , Scored
Hold the CTRL Key for Multiple Selections
2. Choose Viewing Options
Primary Sort
Rank
Ascending
Descending
Secondary Sort
Cumulative Score
Ascending
Descending
Show Report



Exam Summary

No.	Exam Title	Exam Type	Exam Date	Time Taken	Concept Score	Skill Score	Cumulative Score	Rank
1	Chapter 9	Practice	21-3-2010 at 10:17 PM	00:00:57	0 % (0/5)	20 % (2/10)	13 % (2/15)	1/6
2	New Chapter 1	Practice	16-2-2010 at 7:36: PM	00:00:33	60 % (3/5)	0 % (0/0)	60 % (3/5)	1/6
3	Chapter 10	Practice	12-2-2010 at 1:20: PM	00:00:17	70 % (7/10)	0 % (0/0)	70 % (7/10)	1/6
4	Sig Word Specialist Sample	Practice	23-3-2010 at 3:31: PM	00:00:26	0 % (0/10)	20 % (8/40)	16 % (8/50)	1/6

A report is a formatted body of output from a database. It can be generated by request or automatically. This report was created from a database of an online assessment tool.

they are usually issued at the request of commercial firms or consumers. Other types of reports, such as paychecks, are generated automatically on regularly scheduled dates.

Data Filters Some reports can be requested using filtering criteria, called a **data filter**, so that only a subset of the data is presented. For example, if a user wishes to view only accounts receivable overdue by 90 days, a report can be run filtering all accounts except for those overdue by that amount of time. Search engines on the World Wide Web are really data-filtering systems. Search engines maintain massive databases of Web pages. The page contents can be searched using keywords. If a user searches for "cat and dog" for example, a search engine would bring back pages containing the words "cat" and "dog." The report generated in this instance is the HTML Web page transmitted to the user, with the listed Web pages highlighted as blue hyperlinks.

In general, the more specific the data filter, the more limited and focused the result, and the greater the likelihood that the report offers the desired content. Creating good, effective data filters is an important goal of database designers.

Using Databases

A database is ready for data entry and manipulation once it is designed and set up. The activities performed with a database are referred to as **data processing**, which is a broad term describing the handling of various types of interactions or events. Database users perform data processing activities as part of their job.

Data Processing

The processing of database interactions can be set up using batch or transaction processing, or a combination of both of these methods.

Batch Processing With **batch processing**, data processing occurs at a scheduled time, or when a critical point has been reached. Batch processing saves redundant effort by rearranging data all at once, rather than continuously. A database may only incorporate changed records at midnight each day, rather than whenever the new data was entered. Massive databases may take anywhere from several minutes to several hours to update. Batch processing streamlines the process by carrying out the work during off-hours. However, it can introduce some lag into the system, and users are sometimes informed that data will not appear until the day after it was entered. When this occurs, it is usually due to batch processing.

Transactional Processing **Transactional processing** is more continuous and tends to be done with smaller databases or with operational databases that require all information to be very current. A **real-time system**, such as a factory automation or an air traffic control system, can't afford to wait until midnight to update. The data must be kept accurate down to the second in systems that provide data for these types of critical functions. In transactional processing, records of an "event" such as the purchase of an item, the construction of an automobile part, or the departure of a flight are sent to the database for processing one at a time.

E-commerce transactions use **online transactional processing (OLTP)**. These Web sites require fast, always-on processing. One of the great benefits of such systems is that they are effectively in business 24 hours a day. Credit cards are involved in most online sales transactions, and customers expect a quick response so that their orders can be processed.

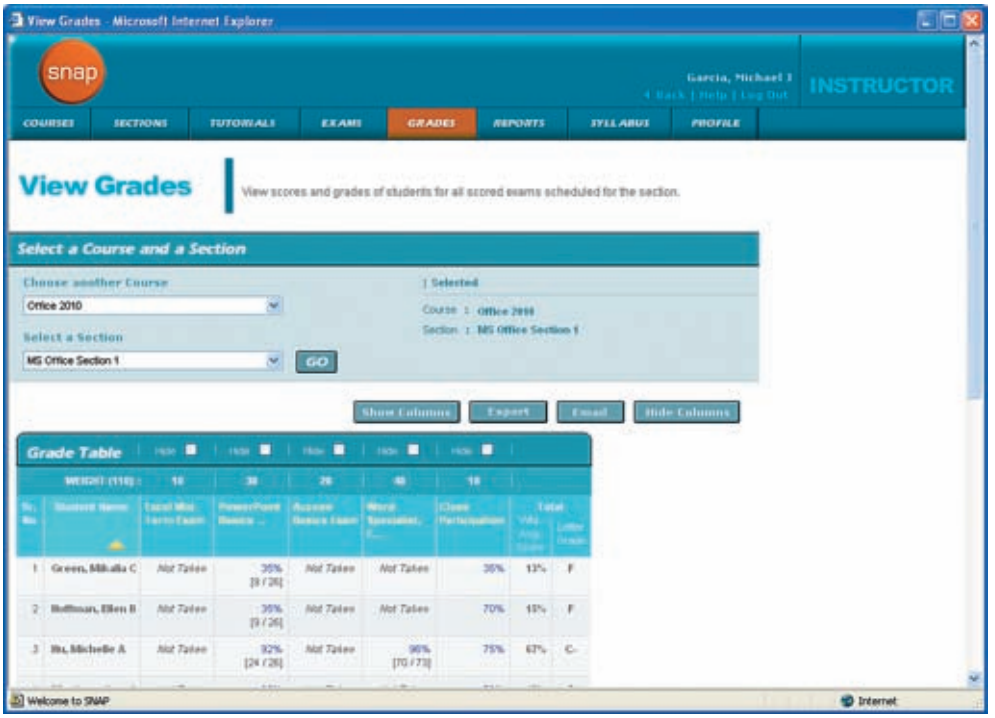
An air traffic control system is an example of a real-time system. Data must be kept accurate and up to date as soon as a change occurs.

Mixed Forms of Processing Transactional and batch processing techniques are often mixed in the same system. For example, in situations involving online orders, a transactional process may be used to handle credit card verifications, while batch processing may be used to handle work orders requesting that items be taken from inventory and delivered to customers.

Database Users

Most people use databases only while performing their jobs, rather than being involved in database design or management. This does not mean that these employees do not have very important jobs. A great deal of effort goes into keeping a database accurate, and this requires constant maintenance. A database must be updated every time a bill is paid, an address is changed, or an order is placed. A **data entry operator** types data into databases and makes sure that it is accurate. Many others work with databases regularly as part of their jobs, including the accountants, executives, and salespeople of any organization.

Many operations are regularly performed on a database as a daily routine. For a situational model, consider a school's grades database. The system controls and maintains student transcripts for all the students of the institution, so records are constantly being added, modified, deleted, and sorted.



A student's grades are entered into a database and stored. The individual grades are used to create a report, or transcript.

Adding Records

Adding records to the system is a common operation. If a student completes a course, a grade record is added to that student's transcript, listing the course taken, the date of completion, and the final grade received.

Modifying Records

If a correction is needed to any prior entry, the record in question could be brought up and the change made on a form, such as a letter grade B being changed to an A. Personal information concerning students might also require changes, such as home and school addresses.

Deleting Records

In the example of a college database, grade records are rarely deleted. Since a grade transcript database is really a data warehouse rather than an operational database, deleting a record does not make sense unless the grades were entered in error and need to be corrected.

Sorting Records

Sorting is a critical function for any database, and involves arranging records in a particular order, such as alphabetically or numerically. When a record is added to a file, for example, the file must be sorted so that the results can be more easily searched. A phone book is an example of a set of printed data that is sorted alphabetically. It is easy to imagine how difficult it would be to find a specific phone number if the listing was not alphabetized. Sorting data in a database (based on the ordering of a key field) greatly reduces the amount of time it takes to find any one record.

Most database systems sort records automatically on the basis of their primary key. Social security numbers are used in many colleges as primary keys, as they are unique and don't require a student to learn a new number. However, sorting records by primary key isn't helpful if the primary key for a student is unknown. If a professor wished to change a grade for a student, but didn't know that student's social security number, it could take a long search to find that one record. In that case the database could be sorted by name, and the results searched to isolate the correct student.

Database Administration

Many factors affect database performance, and thus the quality of the information generated. Database designers must consider each factor, and then ensure that corrections for possible problems are built into the system. Once problems occur, it is the job of the database administrator to solve them. A **database administrator** is responsible for maintaining and updating the database and the DBMS software. Larger systems require a team, or perhaps even an entire database and a department of people dedicated to the task.

Database administrators are responsible for keeping databases up and running to support a company's operations.

Go to this title's Internet Resource Center and read the article titled "Database Administration." www.emcp.net/CUT4e

A database administrator's duties are varied. Any changes in company policy often result in changes to the company database. If billing information sent to customers is supposed to have additional information, the appropriate report must be edited and tested. If a stand-alone system were changed to a networked system, the database must be set up to handle multiple users, and issues such as security and reliability must be thoroughly worked out.

Database administrators are critical players in the success of any organization, since they are largely responsible for preventing computer **downtime**, or time in which the system is unavailable. Consider an airline reservation system. If the computer system is down, the company cannot function. Customers do not like to be kept waiting, or to be told, "The computer system is down." Hearing that phrase usually means that a database administrator is working feverishly to solve a problem. Major factors that database administrators must be aware of include the corruption or loss of data; backup and recovery operations; database response times; record locking; and data integrity, contamination, and validation.

Data Loss or Corruption

Data loss and data corruption are the most serious failures that can occur in a DBMS. Data loss occurs when data input can no longer be retrieved. **Data corruption** occurs when data is unreadable, incomplete, or damaged. Strategies for backing up data are the major method for recovering lost or corrupted data.

Backup and Recovery Operations

A key part of any DBMS is a **backup and recovery plan**. Data can always be lost through power interruptions or equipment failure, so ensuring that data is backed up and recoverable is an important task for database administrators. To lessen the chance of accidental data loss, it is important that backup files are stored separately from original material.

Tape backups are a commonly used backup method, as tapes are cheap mass storage media that can be placed in a company safe for protection. They are common in the financial field because bank databases contain valuable and irreplaceable data. A bank cannot operate if it loses information related to customer accounts. In essence, a bank is a gigantic database, and all of its employees service that database. Since modern banks keep few account records on paper, any database failure would be a catastrophic event.

Banks and other financial institutions commonly store critical data on backup tapes.

Database Response Time

The length of time a database operation takes depends largely on the speed of the hard disk being used. The lag time between a user issuing a command and the database system taking action is called the **database response time**. Network conditions may also affect response time if someone is using a remote access method to perform database operations. This type of delay is magnified in the case of a distributed database, as several computers may have to work together to perform the entire operation. Figure 9-5 illustrates the lag that is often experienced in a distributed database.

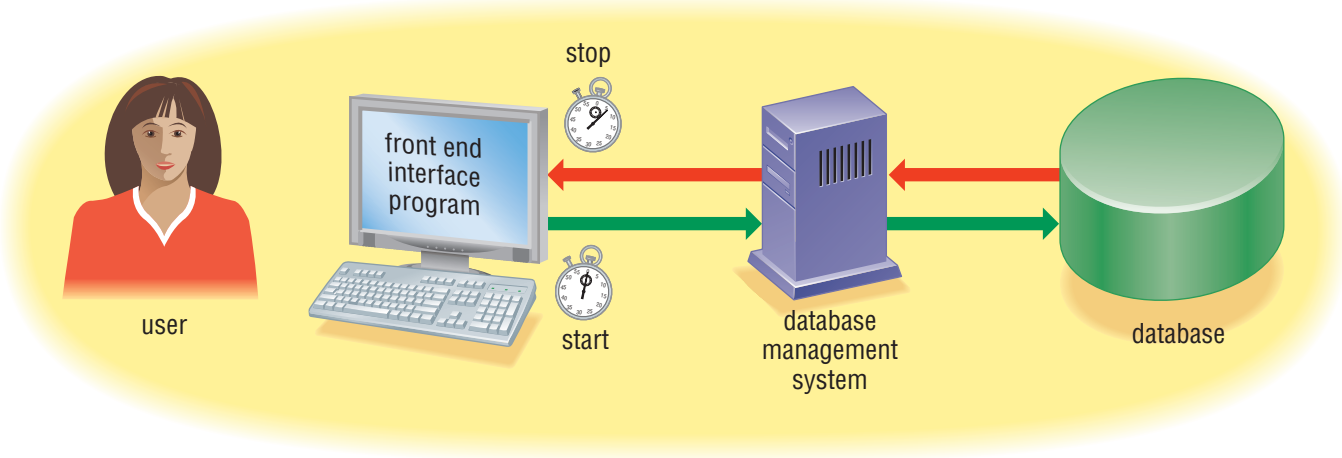


Figure 9-5 Database Response Time
Database response time is the lag time between a user issuing a command and the database system taking action.

Record Locking

Many databases are designed to be used by more than one user at a time. This is usually achieved by networking computers so that information can be shared. Although there is no conflict when two users are viewing the same record, simultaneous attempts to update a record can cause a problem. **Record locking** is an automatic protection process that occurs when users attempt to edit existing records in a multi-user system. The system automatically checks to see if anyone else is working on the same record, and only allows one user at a time to edit or delete the record. During the process of changing a record, it is considered "locked," and the DBMS will generate an error message for any other users attempting to alter it at the same time.

Data Integrity

The term **data integrity** (sometimes called data quality) is used to describe the accuracy of the information provided to database users. A system with high data integrity is obviously more valuable to users than a system containing a large percentage of errors. For example, a sales database with high data integrity will produce mailing lists with fewer incorrect addresses, increasing the percentage of sales material that reaches the intended targets and ultimately increasing sales.

Redundancy, or the duplication of data in several fields, is an enemy of data integrity. Having the same value in multiple places creates opportunities for error when changes are made. For example, if an address is entered in several database locations, any changes to that address would have to be entered in all the locations. Another example of this problem might occur if a person signs up more than once for the same account. Unless there were preventive measures in place, this could cause two separate records to be created. For example, a customer may get partway through an online purchase only to be disconnected. If the customer returns and repeats the process, two copies of the transaction may have been created, leading to duplicate products being delivered, and the customer charged for both.

Redundancy errors are difficult to weed out, and database administrators spend a good deal of time using up-front checks and data validation strategies to locate them. One technique used is called **normalization**, a process intended to eliminate redundancy among fields in relational databases. Normalization works to prevent duplicate data storage, reducing the chance that some data will not be updated when changes are made.

Data Contamination

Once in the system, an error can cause a ripple effect known as **data contamination**. Data contamination is the spread of incorrect information. In many situations data contamination can have serious consequences, such as in credit rating checks. If a company database erroneously shows that a customer has defaulted on a credit purchase, it will eventually be reported to a credit agency. The customer could then receive a poor credit rating, and the company could be liable for any damages resulting from their mistake. Preventing data contamination is one of the goals of data validation.

Data Validation

Among database administrators, the concept of data validation is summed up by the phrase garbage in, garbage out (GIGO). As explained in Chapter 1, GIGO means that bad input will result in bad output, which is why administrators use data validation methods to prevent bad data (garbage) from entering a system.

Data validation is the process of making certain that data entered into the system is both correct and complete. A database is only a reflection of reality, and it is not self-correcting. It is dependent on accurate input to maintain its validity, and therefore its usefulness. Because errors are far more difficult to detect and remove once they are in a system, the best way to ensure that a database is error-free is to prevent errors from being entered in the first place. Anyone who has ever tried to rectify a billing error will appreciate the difficulties involved in straightening things out once they go wrong.

There are a variety of techniques that designers or administrators can use to prevent false data from entering a system. Data validation checks are methods of restricting input so that false data cannot be as easily entered into the system. Validation methods include range, alphanumeric, consistency, and completeness checks.

Referential Integrity

Referential integrity involves a check to make sure that deleting a record in one table will not affect other tables. For example, if you were to delete the accounting department record in your database, then all the employee records that listed a key to the accounting department would have an invalid reference to a record that no longer existed. To make sure this doesn't happen, referential integrity confirms that when you delete a record other records are not depending on it for part of their data to be valid. To continue with our example, in order to delete the accounting department record without an error, you would have to first delete all the employees or move them to a different department before the database would allow the deletion of accounting.

Range Checks

A **range check** is a simple error-checking system usually performed on numeric data entries. For example, to reduce errors in birth date entries, a range check could be created specifying that no birth dates prior to 1890 be accepted, since it is very unlikely that anyone born before that date is still alive.

Alphanumeric Checks

When entering a value for a field, only certain characters may be allowed. An **alphanumeric check** allows only letters of the alphabet and digits to be entered. This would prevent users from entering incorrect characters, such as dollar signs ($), in a "customer home address" field.

Microsoft Office Access
The value you entered isn't valid for this field.
For example, you may have entered text in a numeric field or a number that is larger than the FieldSize setting permits.
OK

Data validation requires complete and accurate input. This Access screen displays if the user tries to enter a null (zero) value in a database field.

Consistency Checks A **consistency check** may be made against previously entered data that has already been validated. For example, a validation system will indicate an error if a user attempts to enter phone numbers or social security numbers that do not match previously validated information for these items.

Completeness Checks A **completeness check** ensures that every required field is filled out. One of the greatest threats to data integrity is the natural human tendency to tire of entering data. This leads users to submit input before every field has been completely filled out. To prevent this, one of the first checks performed on any data entry is a check for completeness, meaning that valid entries were provided for every field marked "required." Incomplete records cannot be allowed into the system, such as credit card numbers without expiration date information.

OnTheHorizon

INDUSTRY OBSERVERS POINT TO SOME NEW TRENDS that could mean more data for the dollar, less work for database administrators, and more efficient systems. These improvements will streamline operations and reduce costs, resulting in savings for customers and clients.

Adaptive Database Management Systems

Database administrators (DBAs) are responsible for maintaining the security, integrity, performance, and functionality of the database systems they manage. Many of these tasks require the hands-on involvement of DBAs, but help is on the way through adaptive database management systems (DBMS) that can learn how to diagnose problems and perform repair and maintenance tasks automatically. If adaptive DBMS progress continues, databases of the future may be entirely self-managing.

Improved File Organization Systems

Efforts are under way to create a new file organization system that bans the simple but tedious file folder directory tree. Newer systems will allow for easier file management with a more intuitive interface. As hard disks bloat into sizes of 100 gigabytes or larger, it is more important than ever to have a system that can locate every file simply and reliably. Newer designs will use virtual reality to let people organize documents and music, audio, and video files in more familiar ways. Imagine storing audio files in a virtual 3D structure like a CD rack, storing family photos in virtual photo albums that are organized on a shelf by date, filing documents in a virtual file drawer with no physical space limitation, and keeping task list items on sticky notes around your virtual house.

New file organization systems will look three-dimensional and resemble everyday settings.

XML Databases

XML (eXtensible Markup Language) is a metalanguage, a type of computer language used to describe other languages. XML has been combined with HTML to produce the next generation of HTML, known as XHTML (eXtensible HyperText Markup Language). The increasing number of documents being written and stored in XML or XML-compliant languages such as XHTML has led to efforts to create and perfect XML databases.

In XML databases, the fundamental database structure consists of XML documents and elements, whereas in traditional relational databases data is stored in tables, organized in rows and columns. Using XML as a database structure means that XML documents can be stored in their entirety, instead of being broken down and stored in the different rows and columns of a relational database. XML documents can then be retrieved in their original format instead of being recreated from stored data.

One current disadvantage of XML databases is that they are memory-intensive, which often makes them slower to process information than relational databases, but this disadvantage is expected to become less of an issue as computer memory becomes increasingly less costly. While XML databases are ideal for working with XML documents, relational databases are more suited to storing numbers and text; thus XML databases will likely complement rather than supplant relational databases in the years to come.

Chapter Summary

For an interactive version of this summary, go to this text's Internet Resource Center at www.emcp.net/CUT4e.
A Spanish version is also available.

What is a database and how does it organize information?

A **database** is a computerized system for storing information in an organized manner so that it can be searched for and retrieved when needed. Data associated with other useful data becomes information. The constant need to alter and amend masses of data is called database maintenance. Databases organize data in a hierarchical fashion, storing information about **entities** in the form of **fields**, **records**, and **database files**.

How do databases help businesses manage information?

The most common database application is an **information system**, which is a system of computer hardware, software, and operating procedures. A **management information system (MIS)** is used to track and control every transaction through a database. A **decision support system (DSS)** is another common form of information system. **Computer-aided manufacturing (CAM)** and **computer-integrated manufacturing (CIM)** are information systems that support factory automation.

How does database management system software work?

Database management system (DBMS) software controls databases. **Keys** are attributes that can be used to identify a set of information, and the designer designates the most important field in a record as the **primary key**. **Data browsing** is the process of moving through a database file, examining each record and looking for information. **Queries** allow users to ask questions designed to retrieve needed information. **Structured Query Language (SQL)** is the most popular database query language. A DBMS also provides **security measures** to protect and safeguard data. **Metadata** is information about data, and a **data dictionary** is the term for a body of metadata. Backup and **recovery** utilities provide a method for backing up and restoring lost data.

What types of databases are available?

Databases are often categorized by the way they organize data (**data models**), or by their function (**operational databases** and **data warehouses**). Common data models include **flat file databases**, **relational databases**, **object-oriented databases**, **multimedia databases**, and **hybrid databases**.

How are databases designed?

The job of planning and designing a database system is usually handled by a **systems analyst**. The development and maintenance of database structures and applications employs a methodology called the **database management approach**. Database designers use **database objects** to build the **front end interface program** and the reporting features. A **form** is a template that allows users to enter data into the database. A **report** is a formatted body of output from a database. Some reports can be requested using filtering criteria, called **data filters**, so that only a subset of the data is presented.

What are the different types of data processing?

The activities performed with a database are referred to as **data processing**. With **batch processing**, data processing occurs at a scheduled time, or when a critical point has been reached. **Transactional processing** is more continuous, and tends to be done with smaller databases or with operational databases that require all information to be current. **Online transactional processing (OLTP)** is used at e-commerce Web sites requiring fast, always-on processing.

How can databases help ensure high-quality information?

A **database administrator** is responsible for maintaining and updating the DBMS software. A key part of any DBMS is a **backup and recovery plan**, to recover data in the event of data loss or **data corruption**. **Tape backups** are a commonly used backup method. The lag time between a user issuing a command and the database system taking action is called **database response time**. **Record locking** is an automatic protection process that occurs when users attempt to edit existing records in a multi-user system. **Data validation** is the process of making certain that data entered into the system is both correct and complete. Validation ensures **data integrity** (or data quality) and protects against **redundancy**. Other validation checks include **range**, **alphanumeric**, **consistency**, and **completeness**.

KeyTerms

Numbers indicate the pages where terms are first cited in the chapter. An alphabetized list of key terms with definitions (in English and Spanish) can be found on the Encore CD that accompanies this book. In addition, these terms and definitions are included in the end-of-book glossary.

Database Basics

database, 400
entity, 401
field, 403
numeric data, 405
alphanumeric data, 405
record, 405
database file, 405

Databases and Information Systems

information system, 407
management information system (MIS), 407
transaction, 407
office information system (OIS) , 408
electronic office, 408
decision support system (DSS), 409
computer-aided manufacturing (CAM), 409
computer-integrated manufacturing (CIM), 409

Database Management System Software

database management system (DBMS) software, 410
key, 410
primary key, 410
data browsing, 411
file processing, 412
query tools, 412
query, 412
Structured Query Language (SQL) , 412
SELECT command, 412
joining, 412
security measures, 412
metadata, 413
data dictionary, 413
legacy database, 414
recovery, 414

Types of Databases

data model, 414
flat file database, 414
relational database, 415
table, 415
tuple, 415
attribute, 415
object-oriented database, 415
multimedia database, 416
hybrid database, 416
operational database, 417
distributed database, 418
data warehouse, 418

Planning and Designing Database Systems

systems analyst, 419
database management approach (database approach), 419
front-end interface program, 420
database object, 421

form, 421
report, 421
data filter, 422

Using Databases
data processing, 422
batch processing, 423
transactional processing, 423
real-time system, 423
online transactional processing (OLTP), 423
data entry operator, 424

Database Administration
database administrator, 425
downtime, 426
data corruption, 426
backup and recovery plan, 427
database response time, 427
record locking, 428
data integrity (data quality), 428
redundancy, 428
normalization, 428
data contamination, 428
data validation, 429
referential integrity, 429
range check, 429
alphanumeric check, 429
consistency check, 430
completeness check, 430

Chapter Exercises

The following chapter exercises, along with new activities and information, are also offered in the Internet Resource Center for this title at www.emcp.net/CUT4e.

Tutorial > Exploring Windows

In Tutorial 9 you learn how to customize some of the display settings in Windows to match your personal needs and preferences.

Expanding Your Knowledge > Articles and Activities

Visit the Internet Resource Center for this title at www.emcp.net/CUT4e, read the articles related to this chapter, and complete the corresponding activities. The article titles include:

- Topic 9-1: The National Electronic Disease Surveillance System (NEDSS)
- Topic 9-2: Relational Database Design
- Topic 9-3: Data Warehousing and Data Mining
- Topic 9-4: Database Administration

Terms Check > Matching

For additional practice, go to the Internet Resource Center for this title at www.emcp.net/CUT4e for a chapter crossword puzzle.

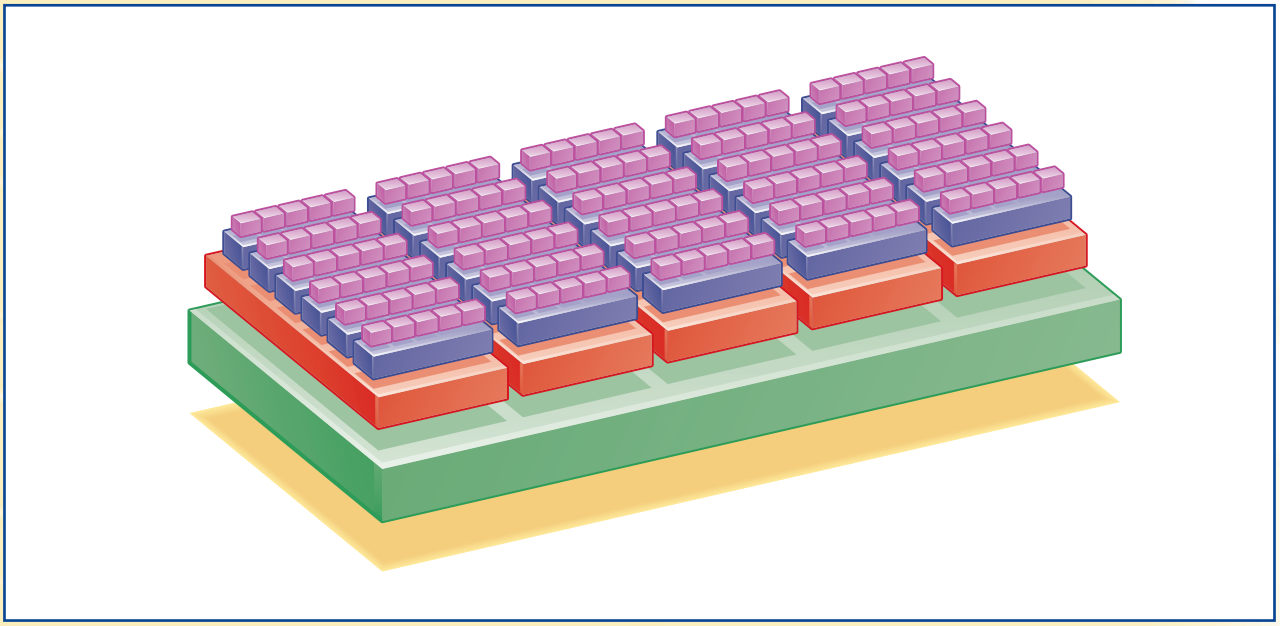
Write the letter of the correct answer on the line before each numbered item.

a. data contamination
b. primary key
c. table
d. multimedia database
e. data validation
f. database management system (DBMS)
g. record locking
h. query
i. field
j. redundancy

____ 1. The smallest element of data in a database.
____ 2. A question posed to a database.
____ 3. In a relational database, the term used to describe what is called a file in other database models.
____ 4. This feature allows only one person to use or modify a record at a time.
____ 5. Errors that are multiplied and carried to other parts of a database, and possibly to other databases.
____ 6. This type of database is used to handle video, graphic, and music files.
____ 7. The field used to uniquely identify each record.
____ 8. Two records containing identical information would be found using this type of check.
____ 9. A set of tools that database designers and administrators use to structure the database system.
____ 10. This is performed to prevent errors from entering a database.

Technology Illustrated > Identify the Process

What process is illustrated in this drawing? Identify the process and write a paragraph describing it.



Knowledge Check > Multiple Choice

Additional quiz questions are available on the Encore CD that accompanies this book as well as on the Internet Resource Center for this title at www.emcp.net/CUT4e.

Circle the letter of the best answer from those provided.

1. Modern information systems are underpinned by
   a. databases.
   b. spreadsheets.
   c. word processors.
   d. operating systems.

2. Primary keys must be
   a. text.
   b. numeric.
   c. unique.
   d. All of the above

3. Which of the following allows users to enter data?
   a. reports
   b. queries
   c. forms
   d. primary keys

4. A(n) _____ is a collection of related fields.
   a. record
   b. database
   c. entry
   d. field

5. Which of the following fields would be the best choice for a primary key?
   a. birth date
   b. first name
   c. last name
   d. Social Security number

6. Microsoft's DBMS is called
   a. Paradox.
   b. SQL Server.
   c. DB2.
   d. Oracle.

7. Querying a large database to identify customer preference trends would be called
   a. maintaining a data dictionary.
   b. data modeling.
   c. data mining.
   d. database administration.

8. SQL stands for
   a. structured query language.
   b. static query language.
   c. statistical quality language.
   d. statistical query language.

9. Which of the following types of information systems would be useful for automating an office?
   a. CIM
   b. OIS
   c. CAM
   d. HTTP

10. Placing records into a logical order using their key fields is called
    a. listing.
    b. organizing.
    c. ordering.
    d. sorting.

Things That Think > Brainstorming New Uses

In groups or individually, contemplate the following questions and develop as many answers as you can.

1. Web databases, such as those search engines use to catalog the Internet, are becoming increasingly intelligent in interpreting queries that nontechnical users write. Go to several search engines that boast of easy-to-use advanced searching techniques and try them out. What other developments do you think may help databases become more intelligent and thus more useful?
2. Smart guns have been developed and are being tested by law enforcement agencies. The guns detect a chip that must be in close proximity to the gun in order for it to fire. Peace officers wear the chip in the form of a wristband or a ring; their weapons will be rendered useless if they are ever wrested away from them. In the future, it is possible that these chips could be injected beneath the skin, making them impossible to remove. If this advance in gun safety becomes commonplace, what advantages or disadvantages do you foresee? Brainstorm possible uses for this technology in other areas.

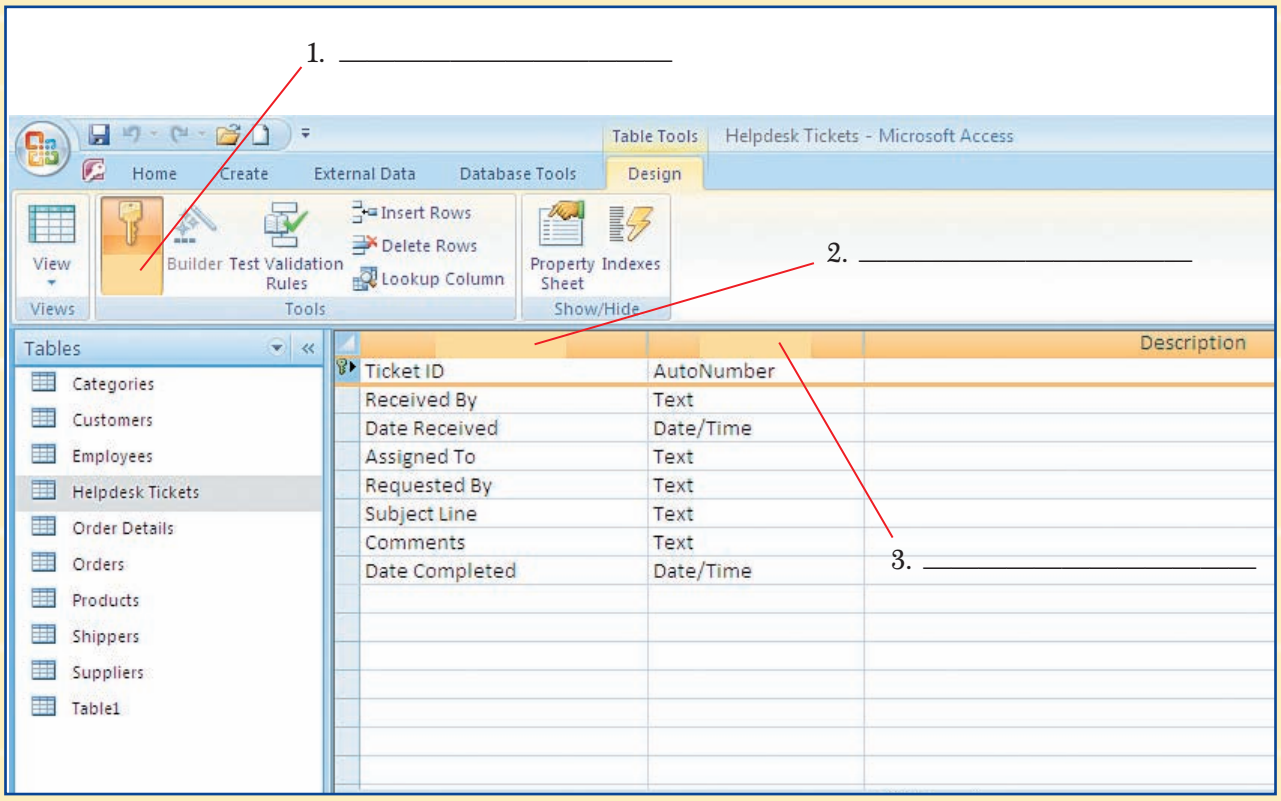
Key Principles > Completion

Fill in the blanks with the appropriate words or phrases.

1. Database administrators typically set up _____ plans to mitigate the consequences of lost data.
2. _____ is a standardized format for entering database queries.
3. When choosing the _____ for a database, the numbers or text must be unique.
4. Information about the data stored in a database is called _____.
5. _____ occurs when data is unreadable, incomplete, or damaged.
6. Matching data from fields in various record files, also called _____, is used when a database request requires data from more than one source.
7. The _____ is the person responsible for maintaining and updating the database and the underlying DBMS software.
8. The accuracy of the information provided to database users is referred to as _____.
9. A(n) _____ is the smallest element of data in a database.
10. _____ systems are slower to respond because all the records in the database must be searched sequentially.

Tech Architecture > Label the Drawing

In this illustration, identify the numbered components of the screen shown below.



Techno Literacy > Research and Writing

Develop appropriate written responses based on your research for each item.

1. What major features does Oracle's DBMS offer? Go to www.oracle.com and study their DBMS products. Do a Web search for other non-Oracle sources of information about Oracle DBMS products. Does Oracle offer any features that its competitors do not? Does Oracle allow prospective customers to download a free demo? Is their software aimed at home PC-type or professional users? Write a summary of the information you gather, including answers to these questions.
2. How would you design a database to describe this course? Create a database outline on paper with the entities that describe this course, including assignments, students, and grading. What types of records would you use? What fields would each record contain? What fields might be shared between records? Make sure that you identify the data type for each field. Do you think your database would be a useful resource for students and instructors if it was made available on a Web page? Would the content of your database pose any security risks?

3. Identify a business that is dependent on databases. Many businesses exist primarily to feed data into a database, process it, and retrieve it as valuable information. Can you think of a company in which practically every employee, and the very product of the business itself, is dependant upon a database? Describe in a written report how each employee interacts with the database, and show how the database forms the core of the overall operation of the organization.

Technology Issues > Team Problem-Solving

In groups, brainstorm possible solutions to the issues presented.

1. In the near future, assume that most people will have access to high-speed Internet connections. You plan to run a dot-com service to provide downloadable music, e-books, and videos to the public for a fee. What kind of database would you use to do this? What kind of fee would you charge—a monthly subscription fee or a per use fee? How much do you think your customers would pay to download music?
2. You have been directed to price a database system for a network of ten workstations. A database license will be needed, along with two laser printers. You have technical staff able to perform the installation and setup, so you only need to price the software license and the printers. Research Web sites such as buy.com and gogocity.com to compare prices.

Mining Data > Internet Research and Reporting

Conduct Internet searches to find the information described in the activities below. Write a brief report summarizing your research results. Be sure to document your sources, using the MLA format (see Chapter 1, page 42, to review MLA style guidelines).

1. It has long been a fear of privacy advocates that personal information will become a commodity, sold on the open market to the highest bidder. Investigate how companies obtain mailing lists and other customer information, and describe the kinds of companies that collect and sell this kind of information.
2. Electronic databases are the rule in most industries, but are not as common in some areas, such as medical records. Investigate the database situation at hospitals and medical facilities in your area. Do they use electronic databases, or are they still in the paper-file era? Are your medical records on the Internet? Do you think they will be some day?
3. With so much personal information stored in various Web databases, the crime of identity theft is on the rise. Research various online sources and write a summary that includes a definition of identity theft, statistics on the prevalence of identity theft, strategies for minimizing the risk of your identity being stolen, and the steps you should take if someone steals your identity.

Technology Timeline > Predicting Next Steps

Look at the timeline below outlining the history of credit cards. Research this topic and try to predict the next logical steps by completing the timeline through 2020.

1914 Western Union gives out metal cards providing credit privileges to preferred customers. These cards come to be known as "metal money."

1924 General Petroleum Corporation issues the first "metal money" for gasoline purchases.

1938 AT&T introduces the "Bell System Credit Card." Airlines and railroads soon reveal their own similar cards.

1959 Many banks offer the option of revolving credit, allowing people to make monthly payments on balances owed, rather than having to pay them at one time.

1966 Fourteen U.S. banks form Interlink, later known as Visa Card, a new association with the ability to exchange information on credit card transactions.

2000 Smartcards grow dramatically in Europe and Asia, becoming the dominant form of electronic payment.

2001 Disposable credit cards good for one-use-only become popular for online consumer purchases.

2003 Orbiscom introduces ControlPay technology, which allows the primary credit card accountholder to control spending limits and locations for various family members.

2004 PayPal reaches 50 million accountholders worldwide for its online payment service.

2008 The average U.S. household carries about $8,700 in credit card debt.

Ethical Dilemmas > Group Discussion and Debate

As a class or within an assigned group, discuss the following ethical dilemma.

Web sites often collect information about a user who has visited a site or has purchased something from the site. Sometimes this is accomplished by placing cookies on the hard drive of a user's computer that can be read the next time the user navigates to that site. Other times, a site requires a user to register some personal information in order to access "premium" features on the Web site. Many users offer this information without thinking about the privacy implications of sharing their personal data.

Because of consumer privacy concerns, more and more Web sites are posting a privacy policy, which typically indicates how they are storing site visitors' personal information, what they do with it, and whether they share it with other parties. However, they may also change this privacy policy from time to time or choose not to follow it. Have you noticed whether Web sites that you frequent have a privacy policy? Would you place more trust in a Web site that has a posted privacy policy? Would you no longer visit a Web site (or purchase from a Web site) if it did not have a privacy policy? Is it illegal or unethical for a company to change its privacy policy without notifying site users? What can you do if a company violates its own privacy policy? How would you recommend that privacy of consumer information be protected on Web sites?

CHAPTER 10

Information Systems

Learning Objectives

- Define information systems
- Explain why information systems are so widely used
- Identify the components of an information system
- Classify different types of information systems
- Discuss the steps in developing new information systems

CyberScenario

JERRY LEANED BACK IN HIS CHAIR and watched his favorite TV show on his computer monitor, keeping the sound down so that it couldn't be heard outside his cubicle. As Jerry sipped his coffee, he thought about how much he loved these mornings. His boss, Rebecca Stinson, was always an hour late on Thursdays. Jerry wasn't too worried about being caught because he had instructed the company's facial recognition system to warn him when Rebecca arrived at the office.

Unfortunately for Jerry, Rebecca arrived at the building entrance at 8:35 a.m., a full 25 minutes early. As she neared the entrance, digital cameras zoomed in on her face from three different angles. Facial recognition software created a three-dimensional model of her face, ran it through a database of authorized employees, and came up with a match. For the rest of the day, almost all doors would open automatically whenever Rebecca approached. But then she was the company's Chief Financial Officer (CFO), so few doors were ever closed to her.

Jerry's computer screen flashed a silent warning before his boss reached the elevator. Jerry scrambled to get things in order as he gulped down the last of his coffee. He quickly closed the video window, replacing the TV show with the purchasing system screen. Jerry immersed himself in matching invoices to purchase orders within the system, trying to prove his work ethic to passersby. He turned around to find his boss watching him. Rebecca said, "Good morning," and Jerry relaxed a bit. Still smiling, Rebecca reminded Jerry that they had a busy day ahead. "No time for games today," she said. Jerry could hear laughter from a nearby cubicle as his boss continued to her corner office. "How could she have known?" Jerry muttered to himself.

Jerry is not the only employee that has used a company computer for personal use. Computers have revolutionized the workplace, but they can have negative effects too. Company information systems drive operations and decision-making, but the Internet can also distract workers with its unlimited entertainment opportunities. Companies often struggle to maintain the productivity levels of workers so they can truly achieve the gains of information systems technology.

The Role of an Information System

An information system is much more than computers or software. Recall from Chapter 9 that an information system (IS) is a combination of people, hardware, software, computer networks, and data that organizations use to manage daily and long-term operations. This can include using computerized information processing for taking orders, tracking inventory, and other transactions. Long-term operations include gathering and organizing the information necessary to make decisions about future growth and other strategic changes (see Figure 10-1).

Fast-food operations offer an excellent example of how information systems function. Cashiers take orders, enter them into computer workstations, and perform monetary transactions. The cash registers are actually sophisticated networked computers. They transmit work orders to the kitchen, where the orders are displayed on

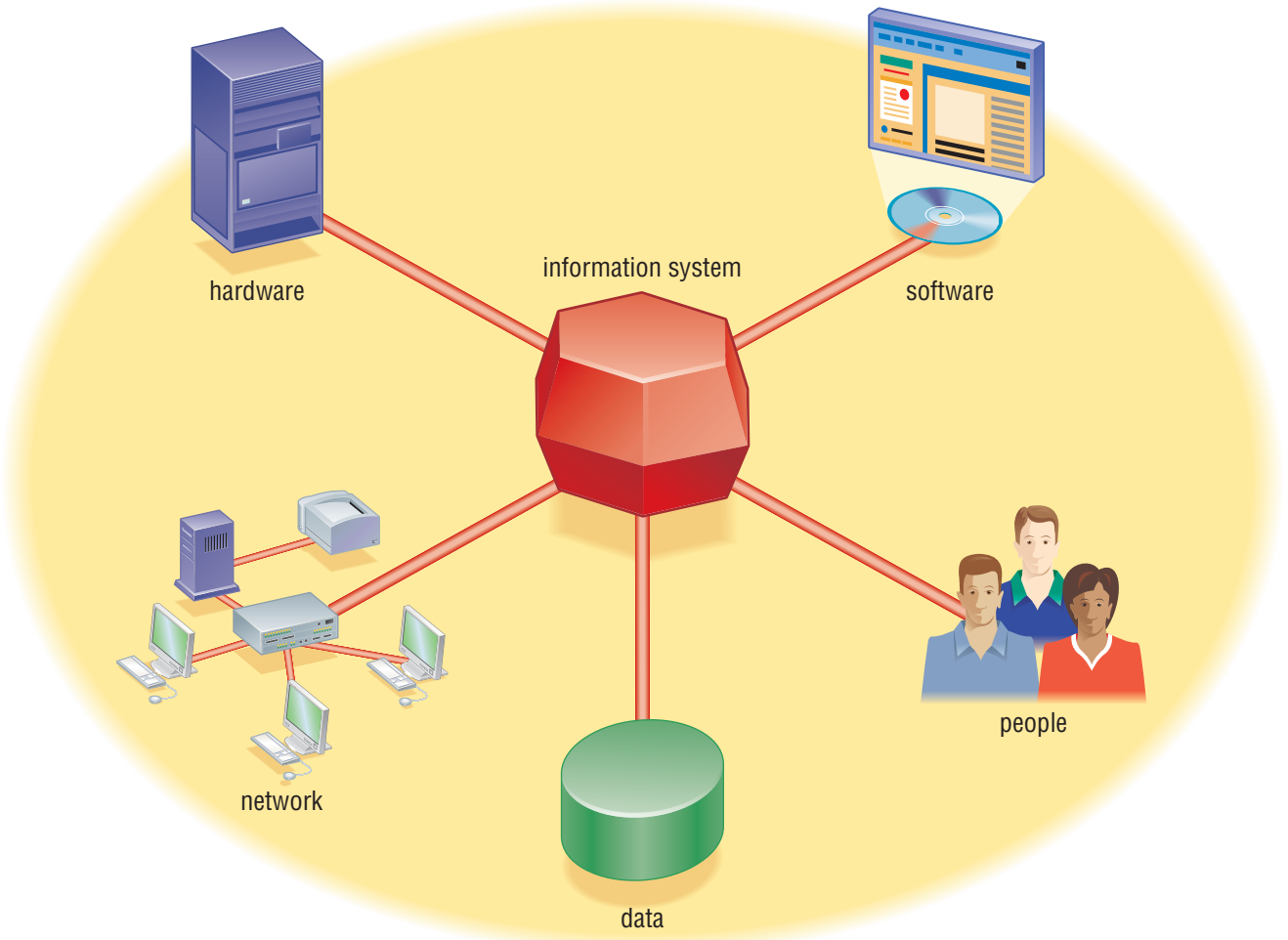


Figure 10-1 Information System
An information system is a combination of people, hardware, software, computer networks, and data that organizations use to manage daily and long-term operations.

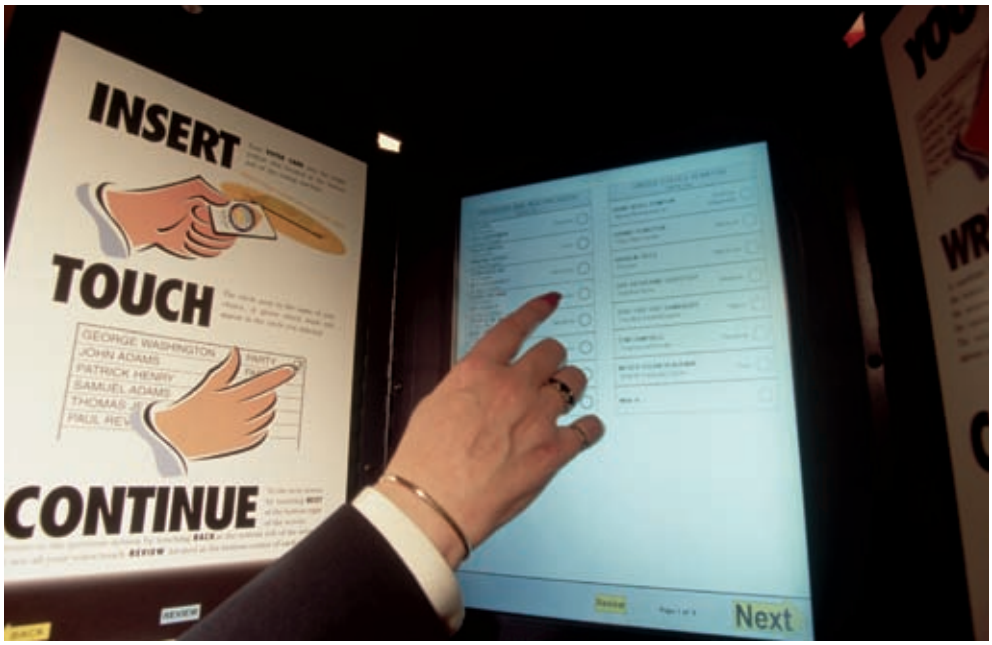
overhead terminals so that the cooks know what food to prepare. The information system database automatically removes items from inventory as they are sold.

The information that the database processes can be used to create reports covering a variety of topics. With a few keystrokes, a restaurant owner can know the day's financial transactions, what items need to be reordered, and which products are selling well. Marketing people in company headquarters can use this information to increase the restaurant's competitiveness by deciding which products should be discontinued, promoted, or possibly sold at a reduced price.

Meeting Business Needs with Information Systems

Why do companies use information systems? The simple answer is that computerized information management increases productivity and efficiency, making companies more competitive. Computer-based information systems offer four key advantages over traditional (precomputer) information systems:

- **improved speed**: Companies are able to perform daily operations very quickly, reducing production times and increasing employee productivity.
- **improved efficiency**: Information systems can be used to design and manage more efficient production processes, saving time and money.



In recent elections, voting machines have been replacing paper systems across the United States. These systems are an example of a new type of distributed information system.

- **improved quality**: Computer-based information systems are less mistake-prone than traditional systems, enhancing product quality.
- **improved measurability**: Computers can be used to track all production factors, making it easier to sort through past data to find trends. This information can then be used to improve products and streamline the production process.

The Rise of Information Systems

The first computerized information systems appeared in the 1950s. They were designed to handle tedious tasks such as sorting mail, printing checks, creating phone bills, and storing demographic data for the U.S. Census Bureau. Some observers worried that computers would take jobs away from people, and in many cases they did. However, new jobs were also created, often requiring more highly trained and better paid individuals. People have now come to accept that computers are here to stay, and that information systems are an inevitable part of their daily lives.

Although we may not realize it, information systems are all around us. ATMs have only been widely available for two decades, but they now rule the consumer-banking world. Gas pumps take credit cards directly; there is no need to go inside the station to pay. Online shopping allows consumers to shop from the comfort of their own homes. At the grocery store, clerks only need to slide a product over a bar code reader to launch a series of important actions—recording the price, deducting the item from inventory, and ordering a replacement. All of these services are examples of information systems in our daily lives. Comparing the speed of these electronic operations with the time they once took to perform manually shows why computerized information systems are so valuable.

Computerized information systems are continually expanding and becoming more comprehensive as they move into business and government activities that humans previously handled. The trend toward consolidation is one of the more striking features of this expansion. Older systems compartmentalized informa-

A gas pump that allows a consumer to insert a credit card, dispense the gasoline, and pay for the transaction automatically is an example of a convenience that is created by an information system.

tion and maintained separate databases for functions such as payroll, billing, sales, and work orders. Newer systems are merging these formerly separate databases, making them more accessible and increasing the number of ways that information can be manipulated. Nothing shows this trend toward consolidation more clearly than the Internet. The Internet is a gigantic information system, linking previously isolated governmental, educational, and business databases. This universal accessibility is one of the biggest factors behind the success of the Internet.

Information Systems Classified by Function

There are many different types of information systems, and the easiest way to differentiate them is by their function. Any system might be used by an individual running a PC or by thousands of employees at a Fortune 500 company. In either case, they provide the same service but on a different scale. The following are some common functional classifications for information systems:

- distribution management
- office information
- management information
- decision support
- executive support
- factory automation

Distribution Management Systems

Getting products to customers probably represents the greatest impact of information systems on business operations. A **distribution management system** is designed to deliver products quicker, cheaper, and with fewer errors. Part of the system is an

One of the uses for an information system is distribution management.

operational database, but the structure also encompasses such seemingly unrelated elements as the duties and schedules of forklift operators and truck drivers.

The major advantage provided by a distribution management system is the reduction of inventory overhead. Traditional systems involved the delivery and warehousing of inventory. The next generation of distribution systems was called just-in-time, and reduced the need to warehouse inventory by delivering it just before it would be used in the manufacturing process. Stockless systems represent the latest type of distribution management system. Savings in inventory overhead have steadily increased as these systems have evolved. The following three distribution systems have different warehouse storage needs, delivery truck needs, and storeroom space needs.

Traditional Distribution Management Figure 10-2 shows an example of a traditional delivery system. Mail-order companies, including publishers, were the typical users of this type of distribution system. Inventory was maintained in bulk stor-

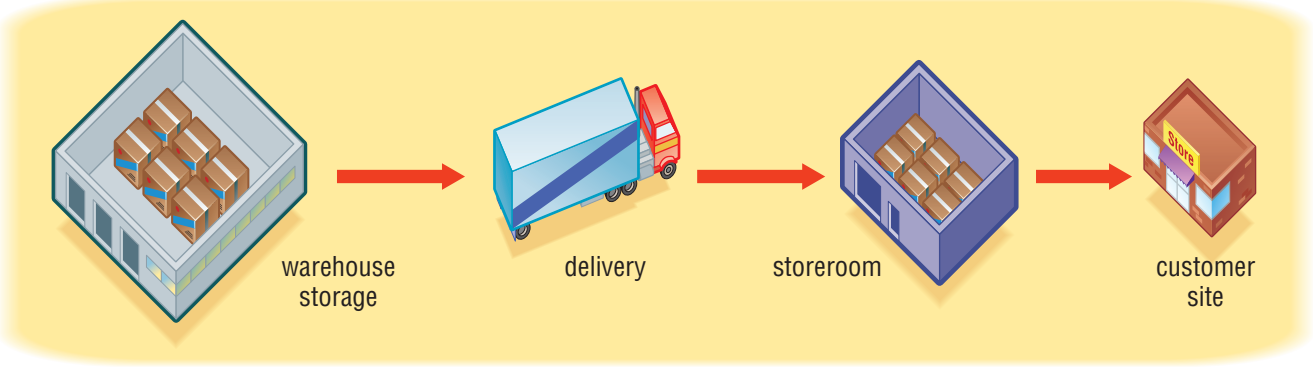


Figure 10-2 Traditional Inventory Distribution System
In a traditional inventory distribution system, regularly scheduled deliveries move products from warehouse storage to storerooms until they are needed to stock the shelves at a retail store.

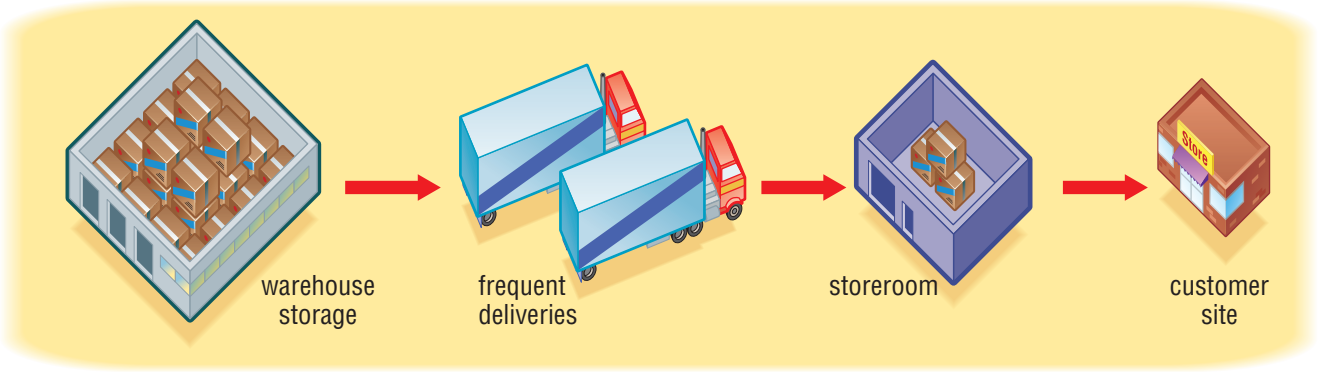


Figure 10-3 **Just-in-Time Inventory Distribution System**
In a just-in-time inventory distribution system, deliveries are smaller and more frequent so that products are only in the storeroom briefly.

age at the place of production, transported to wholesalers in various locations, and finally delivered to retail outlets. This system was slow, and it was often a month or more from the time books were off the presses to the final delivery of the product. This method resulted in large inventories of unsold products, reducing profitability.

Just-in-Time Distribution In recent years companies have begun to focus on reducing inventories and the amount of time it takes to get a product from the factory to the consumer. Called **just-in-time (JIT) distribution**, new computerized distribution management systems allow companies to produce products to match market demand. This shrinks inventories and increases profits. In publishing, this meant a change from large print runs to smaller and more frequent ones, reducing the size of inventories along the entire distribution chain. Figure 10-3 illustrates this model.

Stockless Inventory The most recent trend in distribution systems is a move to eliminate the local storage of products entirely. As shown in Figure 10-4, **stockless inventory distribution** systems work by making daily deliveries directly from the factory, allowing companies to keep their inventory and wasted production to a minimum, while increasing their responsiveness to customer needs. In

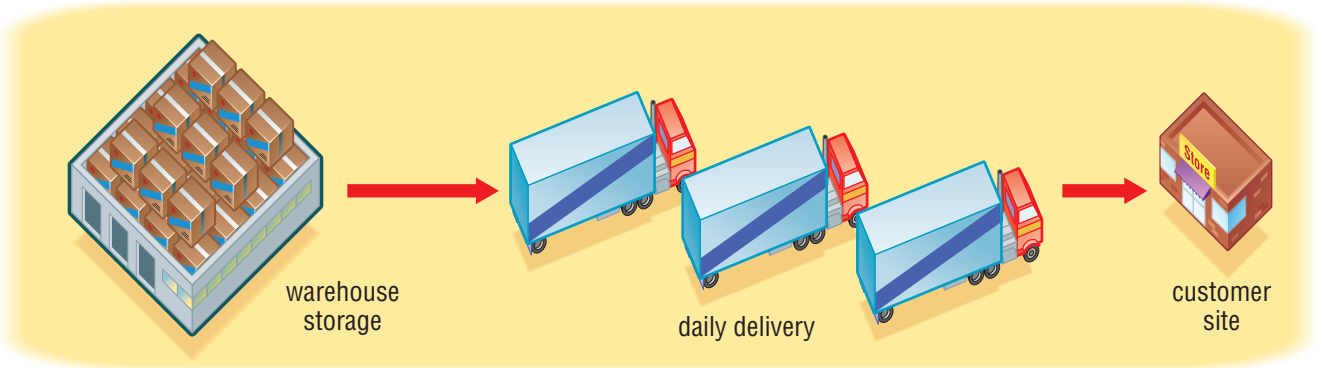


Figure 10-4 **Stockless Inventory Distribution System**
In a stockless inventory distribution system, deliveries move products from warehouse storage directly to the shelves, with no need for a storeroom.

Using computers in a university library or registrar's office, students can access an information system to sign up for classes and plan their schedules for the coming term.

publishing, this includes advances such as print-on-demand (POD), which allows bookstores to instantly print any paperback book from an electronic file, identical in quality to books printed using traditional processes.

Office Information Systems

An **office information system (OIS)** is used to automate office work. These are systems that collect, store, and transmit data in a shared environment. An OIS typically provides the following office automation functions:

- maintaining standard operating procedures
- connectivity
- electronic data processing

Standard Operating Procedures A **standard operating procedure (SOP)** is a set of instructions describing how to perform a task. Workers follow the logical steps contained in SOPs in order to process information correctly. SOPs cover manual and computerized processes, including data entry, generation of reports, and handling of paperwork. In a college setting, there would typically be SOPs for activities such as the admissions process, entering new students into the information system, scholarship processing, financial aid disbursements, and transcript processing.

Connectivity **Connectivity** refers to the ability to link with other programs and devices. Connectivity allows office workers to move information faster and

more accurately from one part of the information system to the next, without documents being lost or forgotten along the way. Connectivity is enhanced by computer networks that facilitate faxes, e-mail, and other forms of electronic document transmission.

Electronic Data Processing **Electronic data processing (EDP)** is the use of computers to process data automatically. This function can take a variety of forms, from the creation of new documents to the production of paychecks using a payroll processing system. Transactional processing is used for most situations involving standardized data entry, such as fast-food restaurant ordering systems. Businesses handling large amounts of data input often use batch processing. For example, the banking industry uses batch processing to process the millions of checks and cash card transactions that need clearing every day. This is usually more efficient than updating the entire database every time a change is made.

Management Information Systems

With the exception of very small companies, nearly every business uses some form of a **management information system (MIS)**. As the name suggests, an MIS exists to provide information that assists in the management of business operations. Management information systems typically integrate information from different company departments or functions into one database. It is this combining characteristic that enables the comprehensive reporting capabilities of an MIS. An MIS usually provides support to management in the form of regular reports, either on a periodic or an on-demand basis. Quarterly earnings, monthly cash flow, and other types of reports provide important information that helps managers improve productivity and profitability.

Automated teller machines (ATMs) are part of a bank's information system. They provide operational access to end-users.

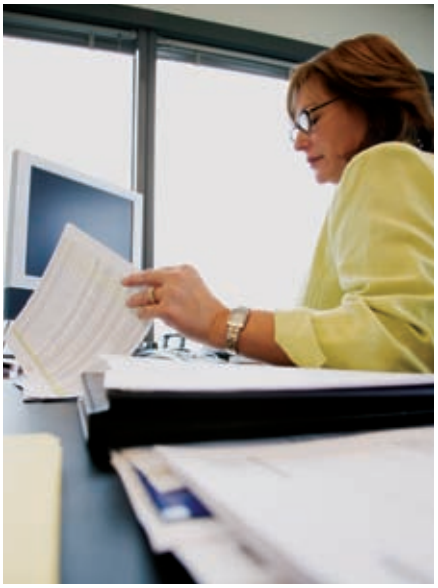
VICE PRESIDENT, SUN MICROSYSTEMS

James Gosling

JAMES GOSLING'S NAME IS PROMINENT on the list of famous programmers. A Sun Microsystems fellow and vice-president, Gosling is best known as the creator of Java.

James Gosling wasn't trying to revolutionize the computer industry when he created the Java programming language. His object-oriented language is based on the power of networks and the idea that the same software should run on many different kinds of computers, consumer gadgets, and other technology devices. Since its initial commercial release in 1995, Java technology has grown in popularity and usage because of its true portability. It is a platform that allows users to run applications on any compatible device that supports the Java platform, allowing programmers and users to do new things with Web pages that were never before possible. Before his arrival at Sun Microsystems in 1984, he built a multiprocessor version of Unix, multiple compilers, and mail systems. In addition, he also built the original "Emacs," a text and drawing editor for Unix systems, and helped to build a satellite data acquisition system. While earning a PhD in computer science from Carnegie-Mellon University in 1983, he worked as a principal on the Andrew Project, a high-speed and wireless infrastructure, now one of the largest installations of its type. His early activity at Sun as lead engineer of the NeWS window system led to the original design of the Java programming language and the implementation of its original compiler and virtual machine.

A decision support system (DSS) is a type of MIS that provides information that allows managers to review and improve a company's profitability.



Decision Support Systems

A **decision support system (DSS)** helps managers make informed business judgments. A DSS is generally custom-built for each application. Spreadsheet programs and types of artificial intelligence are often used to create a new DSS. A DSS might exist independently of an OIS or MIS, but it is usually provided by the same software and database system.

Decision support systems do more than track input data and print summary reports covering what happened during the previous month. One additional feature is called data modeling. **Data modeling** provides a

Gosling did not develop Java as an individual pet project; its original name was "Oak," and it was part of a secret mission called "The Green Project." Gosling and a group of Sun employees were chartered to figure out the next "wave" of digitally controlled consumer devices and computers. Gosling recalls, "As we were building stuff, it became clear that one of the obstacles to the whole endeavor was the programming language and the tools we used to build the software. So my part of the project was to go off and solve the tool problem, and that's where Java came from. The vision was to build networks of big and little things connected together with Java as the concrete and nails."

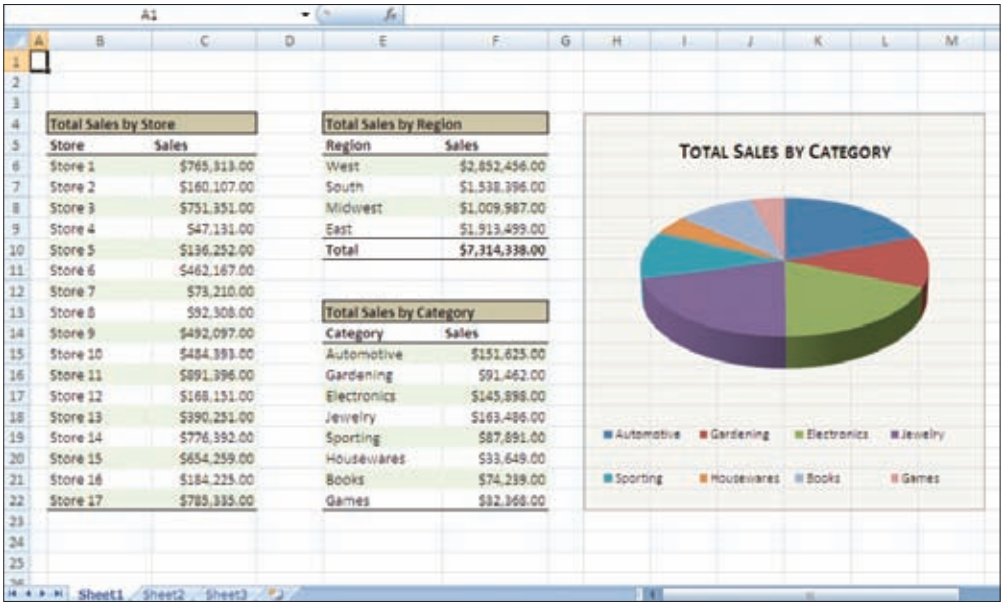
A driving force underlying Java's origin was Gosling's belief that technology and business go hand in hand through digital circuitry. Java became the obvious component to enable that merger. Sun Microsystem's recent push is to put Java 2 Micro Edition (J2ME) into wireless headsets, allowing users to dynamically access and respond to Web content such as e-mail or stock quotes. For the future, Gosling envisions a plethora of wireless-embedded devices and the intersection of Java and cell phones.

In an interview with *Wired News*, Gosling maintains that his creation of Java started out as "something done for me and friends. It's almost a historical accident that I ended up doing what I did. We had no illusion of becoming widespread, and no worry about any particular goal. What's happened since then is nutty!" Gosling recalls that he had no idea the language his team was hammering away at would become an industry buzzword. "It was more of a matter of entertainment than creating a product."

Recently, Gosling spends most of his time concentrating on developer tools that will meet the needs and speeds of faster processors and more complex systems. As the Chief Technology Officer at Sun Microsystems, he is questioning how to use tools to help ease the use of these complex systems.

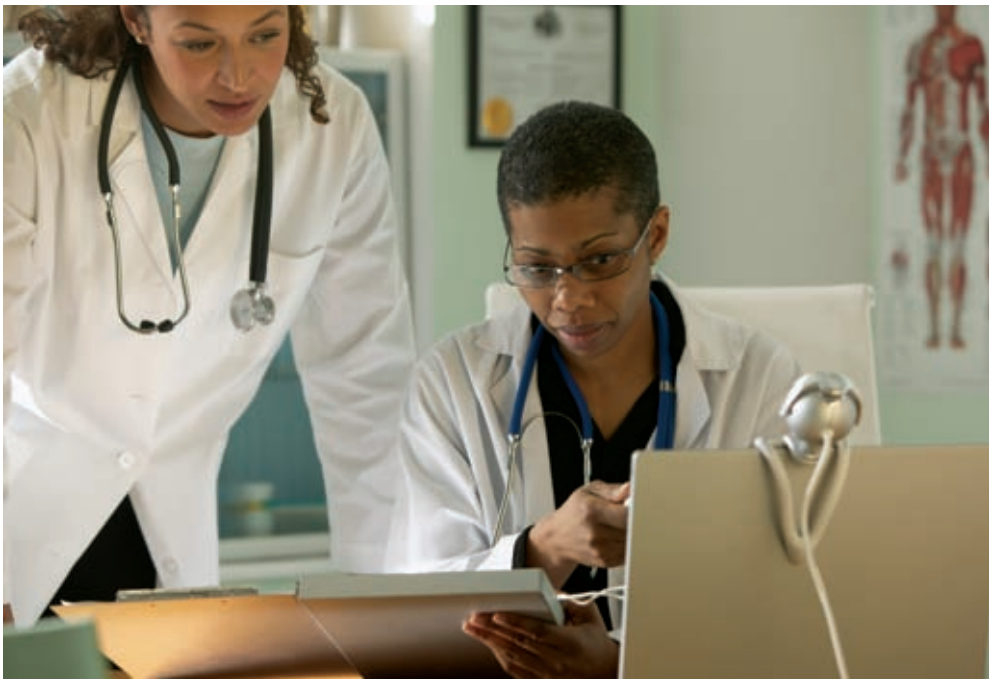
simulation of a real-world situation built into a software application. With data modeling systems, users can input numeric data into "what-if" scenarios to predict future outcomes. For example, company leaders can use data modeling to help them decide whether to open a new branch office or to determine loan payments at different rates of interest. The ability to visualize alternative scenarios is a valuable decision-making aid.

Spreadsheets Used to Create DSS Applications One of the key drivers of the personal computer revolution in the early 1980s was the popularity of spreadsheets as simple decision support systems. The spreadsheet is widely recognized as the "killer application" that drove early PC sales into the millions. People were using spreadsheets for "running the numbers," business jargon for performing what-if analyses on a DSS. Many people have convinced managers and lending institutions that their business proposals were sound investments using nothing but a PC and a spreadsheet application.



A spreadsheet program such as Microsoft Excel can calculate elements of a business proposal, such as sales history, and display them in a chart.

Expert Systems An **expert system** is a sophisticated DSS that attempts to model an expert's knowledge of a topic. An expert system contains a set of facts and rules about a specific, complex application called a **knowledge base**. The system will make recommendations after posing a series of focused questions to the user. Expert systems exist to aid doctors in diagnosing rare diseases, to help geologists find oil and mineral deposits, and to troubleshoot network connections. They qualify as decision support systems, but they are also a major component of the field of artificial intelligence. For more information concerning expert systems, see the section on artificial intelligence in Chapter 13.



A doctor might use an expert system to help diagnose a rare disease.

Online Analytical Processing An **online analytical processing (OLAP) system** defines and clarifies methods of handling what-if questions posed to large databases, as well as to massively distributed data systems such as the Internet. Originating from a "white paper" commissioned by Arbor Software and written by Dr. E. F. Codd in 1993, this new software class has fostered the development of a variety of software packages. The field has yet to completely evolve, but the major thrust of the technology is focused on providing better ways to analyze the mass of data now pouring into databases, thus producing more useful results. The information gained from the effective use of OLAP systems is called **business intelligence (BI)**. Some examples of business intelligence large corporations use include customer spending patterns and behavior, competitive positioning with other companies, and risk factors for credit or insurance purposes.

Executive Support Systems

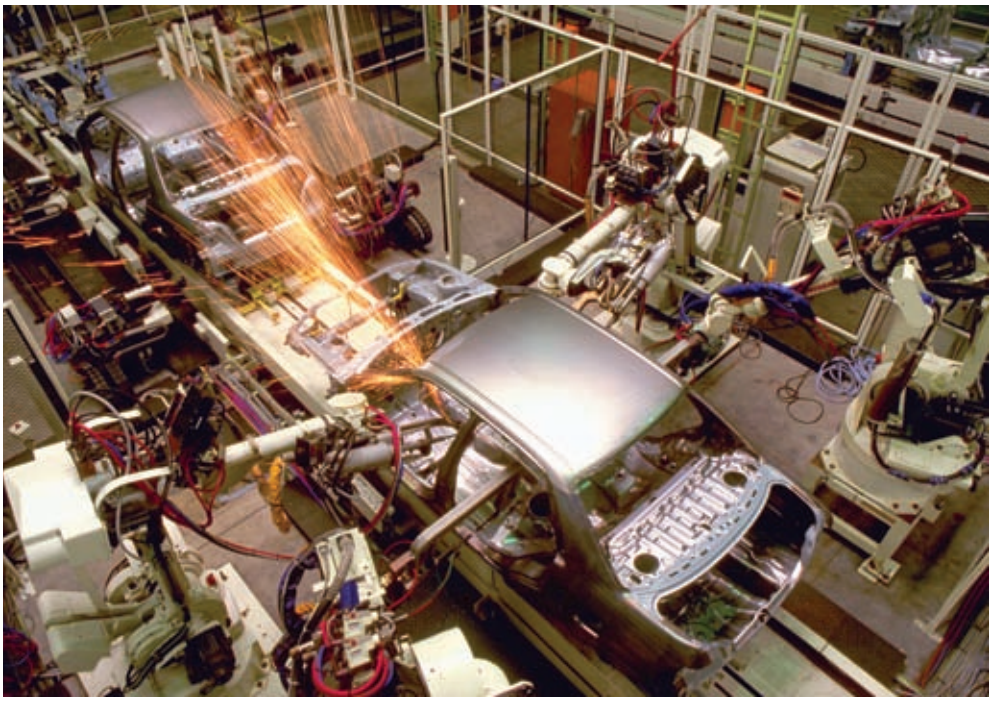
An **executive support system (ESS)** functions as an upper-management tool. It is similar to a decision support system but is generally more intuitive and broader in focus. Executive support systems attempt to bring together information from an entire organization, rather than focusing on any one department or group. An ESS uses this information to support strategic decision-making by facilitating the creation of high-level summary reports that executives can use to help make any number of important decisions. For example, an ESS is often used to handle accounting issues for corporations, such as cash flow, quarterly earnings, and dividends. In addition to knowing which burgers sell best, the CEO of a fast-food chain might want to know which locations are experiencing the best sales growth. This information could then be used to maximize profits by ensuring that outlets are only opened in the high-growth locations the ESS identifies.

Factory Automation and Plant Operations

In addition to being used in offices, information systems are frequently found in factories. Two major systems are used in factory automation: computer-aided manufacturing (CAM) and computer-integrated manufacturing (CIM). Statistical quality control (SQC) is another form of information system often used in plant operations.

Computer-Aided Manufacturing Recall from Chapter 9 that a **computer-aided manufacturing (CAM)** system involves the direct use of computerized systems to manufacture products. Robots used in automobile manufacturing plants are a classic example of this technology. Even more common are systems that monitor and report on the progress of an assembly line. These CAM systems provide factory operators with information such as temperature readings, assembly line speeds, and quality test failure statistics. Any computerized system that directly aids in a manufacturing process is called a CAM system.

Computer-Integrated Manufacturing A **computer-integrated manufacturing (CIM)** system offers much broader coverage than a CAM system. CIM systems focus not only on the factory floor, but also on the company as a whole. A CIM system connects the factory floor to the executive offices, the accounting department, and the sales staff, making information available to anyone who might need it. A



Automobile manufacturing plants commonly use robots as part of computer-aided manufacturing (CAM) systems.

Manufacturers of all types use computer-integrated manufacturing (CIM) systems.

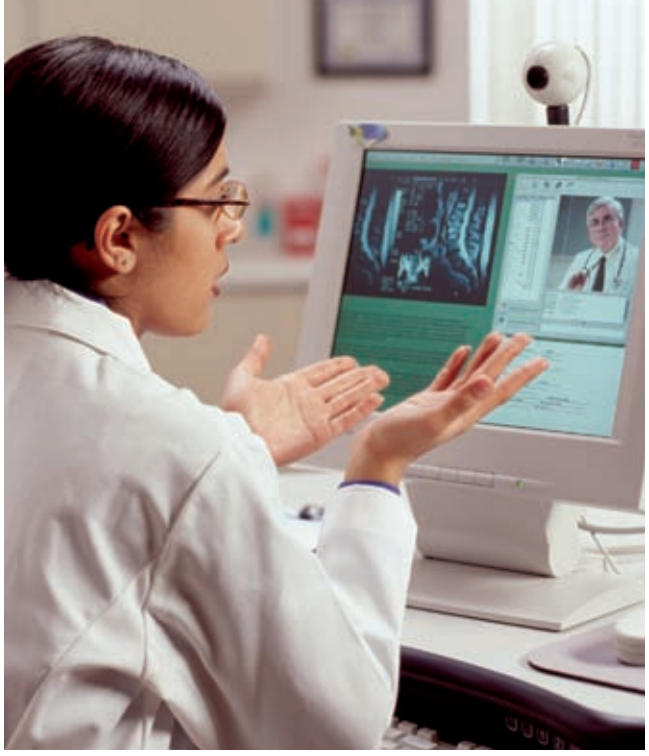
much larger form of information system than a CAM, CIM systems impact everyone in a manufacturing company, from the president down to the mail clerks.

In an automotive company employing CIM, factory lines would use individual CAM systems to help operators monitor and control production. These CAM systems would be networked to a CIM that tracks production progress, errors, and quotas. This information would be automatically sent to headquarters for evaluation. Orders would also be sent from the management information system telling the CAM lines what to produce and in what quantities.

Statistical Quality Control A **statistical quality control (SQC) system** uses a combination of data tracking and data modeling to build a picture of how well a factory is operating. An SQC tracks measurements of the quality of a final product and compares them with the conditions present throughout the manufacturing process. For example, through the use of an SQC, a car manufacturer may discover that some supplied metal caused more defective parts than other metal or that one shift of workers is generating better quality products than another group.

Considerations for Planning and Updating Information Systems

Planning or updating an information system is a process that organizations initiate in response to a variety of factors. Management may decide to "computerize" paper-based systems that worked well in the past but are no longer capable of keeping up with today's marketplace demands. In other cases, a company may have a computer system and databases that are accessible only to a few people with specialized skills. A new system may become necessary to accommodate the expansion of the business. For example, a small professional office might join with several other offices to form a larger practice. Its billing and receiving information may be on one computer database, while its payroll and inventory systems may still reside on paper forms. The larger practice could utilize a new information system to integrate all the functions and records of its member offices, allowing access to the practice's data from any workstation.



A well-designed information system can make resources and information more accessible. For example, a physician in an office can view a patient's medical record while another physician treats the patient in the hospital.

Impact of a New Information System

The impact of a new information system should be studied in order to determine its suitability. Replacing a traditional paper system can result in a difficult adjustment process for some employees. Sometimes a major information system upgrade has almost as dramatic an effect as did the switch from a paper to an electronic system. The major effects of upgrading or switching to a computer-based information system are:

- expanded technical staff
- increased information technology costs
- increased profit margins
- enhanced product quality
- reduced clerical staff
- temporary disruption

While the impact of a new information system has both positive and negative effects, the negative effects are usually short term. Information systems should be perceived as investments in the future, and therefore they may take time to return their full value.

Expanded Technical Staff In the short term, new technology requires additional technical personnel, such as technicians and programmers. The transition period can last a year or more at large companies, and the company payroll will increase dramatically. In addition to paying existing clerical staff to keep business rolling, the company must pay new technical staff to retrain the current staff so that they can operate the new system.

Information technology costs are high because computers need to be upgraded continually, making it necessary to consider those expenses when businesses create their annual budgets.

Increased Information Technology Costs

Computers are not like chairs or desks. They do not last for decades; they normally last only a few years before they are obsolete. Since there is a constant annual cost involved in keeping up with the latest technology, computers should be viewed as short-term purchases rather than fixed-cost investments. When pricing new systems, a common mistake of many managers is to forget to account for the cost of the inevitable upgrades that will be required in the years following the initial purchase.

Increased Profit Margins The big payoff of any information system is the ability to do business faster. This takes time to materialize, and new systems usually aren't cost-effective for the first few years. Information systems may not always reduce a company's bottom line, but they do enable a company to grow and do more business than it would have without a system in place.

Enhanced Product Quality Once an information system is fully functional, improvements in product quality should follow. For example, a company that manufactures computer printers should see several improvements by implementing an information system. Supplies will arrive more quickly and with fewer errors in the orders. If the information system includes quality elements, products will be superior to those produced using regular systems. Higher-quality products mean more satisfied customers and therefore increased sales—the ultimate goal of any business change.

With a well-designed information system, all aspects of product production and delivery are monitored. As a result, items remain in stock and available for customer purchase.

Reduced Clerical Staff One of management's goals in installing or upgrading an information system is to reduce clerical staff positions. As a general rule, computerized billing, order taking, and credit approval processes translate into a reduced need for personnel. Since employee salaries typically represent a company's largest single monthly expense, an efficient information system can cut costs significantly. However, the costs of additional computer hardware, software, and more technically capable employees cut into any savings.

Temporary Disruption Another factor that people often fail to take into account is that operations will almost certainly experience disruptions. On the technical side, hardware systems take time to perfect, and errors will emerge as the system is implemented. On the human side, switching systems is stressful for staff and management alike, as few people enjoy having to change the way they are used to doing things. Employees become comfortable with their jobs and they tend to avoid change, particularly the significant changes brought about by new computer-based information systems.

Internal Development vs. Outsourcing

Once the impact of a new system has been studied and a decision to proceed has been made, a new system must be purchased "off the shelf," or developed and brought into service. The buy-or-develop decision is influenced heavily by the size of the company. Larger organizations usually have in-house IT staffs devoted to developing and maintaining information systems. Smaller organizations usually can't afford to pay employees to develop information systems, so they end up purchasing standard software packages, or subcontracting their

Some organizations will outsource the development and maintainence of information systems to a systems integrator as a more cost-effective way to meet a company's technology needs.

Home Trackers

DID YOU KNOW THE GPS COORDINATES for the White House are 38.898648N latitude and 77.037692W longitude? Just as you know the exact location of the White House, the U.S. government will soon know the Global Positional System (GPS) coordinates of your front door within 40 feet!

Supported in part by a $700 million taxpayer-funded contract, close to 140,000 workers are busily collecting GPS coordinate data for every front door in the U.S in preparation for the 2010 census.

While the scope and technology of this project are no doubt astounding, many people are questioning why the information is needed. Critics of the program argue that the information census takers collect will make it easier for the government to monitor you, and that it has nothing to do with the census.

The Census Bureau argues that the information will make it easier for census workers to locate addresses, will ensure residents are counted in the correct location, and will greatly reduce the number of geographic coding errors caused by using paper maps in previous counts. Other proponents of the program suggest that in the event of martial law or foreign troops helping out on U.S. soil, emergency teams not familiar with our language, street signs, or general lay of the land could use the GPS coordinates to locate those in need.

Whether you question the potential for Big Brother to use your GPS information for evil, or you side with the Census Bureau's and U.S. government's argument that the data will help ensure good census data and aid emergency responders, you cannot deny that the technology is impressive.

information system development and support work. Often called **outsourcing**, this practice involves hiring a third party to handle the project, usually a consultant or a **systems integrator**. A systems integrator is a company that specializes in installing and supporting information systems.

As a company grows and its needs become more complicated, hiring a technician for system maintenance and support may become a necessity, since hourly consulting rates are much higher than a technician's salary calculated on an hourly basis. The amount paid to a consultant for one hour's work may be the same as the amount paid to an employee for a full day.

INTERNET RESOURCE center

Go to this title's Internet Resource Center and read the article titled "Value-Added Resellers and Original Equipment Manufacturers." www.emcp.net/CUT4e

Factors to Consider When Choosing a New Information System

Deciding which information system to purchase is difficult, particularly for small organizations. In the early planning stages, managers may have only limited knowledge to guide them. It is not uncommon for nontechnical people to find themselves in charge of deciding which hardware and software combination they should purchase—without having any idea about how to make that decision. The following factors should be considered when choosing an information systems package:

- user interface design
- open versus closed system
- niche information system
- turnkey system
- system support

Go to this title's Internet Resource Center and read the article titled "Horizontal Market vs. Vertical Market Software." www.emcp.net/CUT4e

User Interface Design

One of the first things to examine is the system's user interface. The interface design should be user-friendly and intuitive. Critical information should be readily apparent. It is wise to examine a number of different interfaces before making this judgment. Information system software is expensive, and mistakes will be costly. Generally, client/server and Web-based applications are more intuitive and user-friendly than mainframe applications. However, each software package is different, so it is important to assess systems thoroughly.

Open or Closed System?

Choosing between open- or closed-system formats is one of the most crucial decisions. In an **open system**, the company's IT staff can alter the system or hire a third party to take care of any necessary alterations. On the other hand, a **closed system** must be serviced and supported by the original vendor. Because of this requirement, closed-system support services command a price that can surprise inexperienced buyers.

Niche Information System

A **niche information system** exists to serve specially focused sets of customers. This type of package is also commonly referred to as a vertical market package. An example of a niche system would be a package designed for an auto-body repair shop. The system would include all needed forms and templates, such as those for invoices, customer service records, and insurance claims. A major benefit of this type of information system is that it will serve an organization's needs well, without requiring a lot of technical know-how or staff training. The market for these specialized information systems is small, which means the price per system can be quite high. Another drawback is that vertical market packages often tend to lag technologically, and they may not be compatible with the latest operating systems.

Niche information systems are designed for specific applications, such as running an auto-body repair shop.

Turnkey System

A **turnkey system** contains everything a business needs to get a new information system up and running. The name comes from the idea of being able to simply "turn the key," similary to how you start a car by simply turning the key to start the engine. There should be no complications, and no technical skills are required. Turnkey systems are usually the best and the most expensive, as they are heavily customized and include the services of support people to tailor them to the exact needs of a business. For this reason, they usually are out of the price range of small businesses. Like any custom-built product, they tend to be a bit inflexible. It is very

IT Under the Big Top

BACK IN 1984 a group of street performers in Montreal banded together to form a show. In years since, this small troupe of jugglers, stilt-walkers, and fire-eaters has grown into a mind-blowing spectacle of music and movement, the world-famous Cirque du Soleil.

The circus company now has eight shows touring the globe, in addition to ten on set stages. Its extensive operations demand a technological approach to monitoring the artistic efforts that go into each performance.

The company's online database, called Cirque Memory, is accessible in five languages and covers six applications. The casting directory tracks 20,000 performers and their careers; a make-up application contains detailed photos and instructions; and a costume database catalogs costume sizes, colors, locations, and alteration notes. The Medi-Cirque file tracks employee health and rehabilitation programs. Kin-cirque is the program that develops and tracks fitness training for each artist and act. Finally, the Act Management program allows trainers to follow each precise moment in a specific act.

To the more than 40 million people who have viewed the Cirque du Soleil, it is an entertainment triumph of seamless music, color, and talent. But the behind-the-scenes information system that tracks and coordinates the innumerable creative details also deserves some applause.

expensive to upgrade turnkey systems to take advantage of any improvements when the underlying technology changes.

System Support

The level of technical support is a key issue, whatever the system under consideration. Questions to be considered include whether or not installation, staff training, and on-site service and repair are included in the system package. If not, the cost of these services needs to be investigated and factored into any calculation of the total cost. All too often a call for technical assistance is likely to end up in a voice mail / holding pattern for hours, with the caller paying a per-minute charge for assistance. The need for technical support and maintenance is a given, so it is important that the level of support be a known quantity before purchasers sign on the dotted line.

INTERNET RESOURCE center

Go to this title's Internet Resource Center and read the article titled "Support Contracts." www.emcp.net/CUT4e

The Process of Developing a New Information System

Identifying and assembling employees with the skills and expertise required to develop a new in-house information system is a necessary first step. A **business systems analyst** is involved in the early phases of systems development. Through interviews with management and users, the analyst defines the business problem and plans a solution. In response to the business systems analyst's recommendations, software engineers and programmers handle the design and implementation of the new system.

Most technical support is provided over the phone by customer service representatives and specially trained technicians.

Programmers specialize in the development of new software, while software engineers are highly skilled professionals with programming and teamwork training. Their organized, professional application of the software development process is called **software engineering**.

Project Team

The implementation of an information system necessitates the creation of a project team. A project team usually includes a project manager, who acts as the team leader. Sometimes the project manager also functions as a systems analyst, responsible for completing the systems analysis and making design recommendations. The rest of the project team includes software engineers and technicians. The software engineers deal with programming software, while technicians handle hardware issues.

The comprehensive process software engineers initiate is called the **system development life cycle (SDLC)**, a series of steps culminating in a completed information system.

INTERNET RESOURCE center

Go to this title's Internet Resource Center and read the article titled "The Buy vs. Build Decision." www.emcp.net/CUT4e

System Development Life Cycle

Creating a development plan and executing it has several stages, some of which have already been set in motion as part of the decision to proceed with the project. As shown in Figure 10-5, the process includes planning, design, implementation, and support, but always includes returning to the planning stage, to continue through the cycle again. The planning stage includes preparing a needs analysis, conducting a feasibility study or studies, establishing a project team, and creating a project plan. In the design stage, the team develops the functional specifications and the design specifications. The implementation stage includes designing a prototype,

Systems development requires the ability to work well in a team environment.

developing the system, purchasing hardware and software, installing and testing the system, training the users, and approving the system. The support stage includes identifying technical support requirements and providing a system for updates.

Planning a System

To understand how the information system development process works, imagine a scenario involving an automobile manufacturing company named Big Engines, Inc. Big Engines' management decides that their old system isn't working well, and they see an opportunity to improve their current operations. After recognizing the need for a new system, they can either contact their company's own internal development team (if one exists) or they can contact an outside vendor.

Suppose Big Engines is considering an upgrade to their existing CIM. A **feasibility study** would be conducted to investigate how difficult the project might be to complete and how much it might cost. A large automobile company like Big Engines is almost certain to have its own internal development team. However, the internal team may recommend hiring a systems house if the project is beyond their capabilities, or if the team is too busy handling existing problems.

If management decides to hire a systems house, a **request for proposal (RFP)** will be sent to one or more possible suppliers. The systems houses would then send representatives to determine what is required before quoting a price. In the case of larger, more expensive projects, systems houses might be asked to bid on the job, meaning that they must examine the proposal thoroughly before providing an estimated price for completion of the system. The lowest qualifying bidder meeting with company approval gets the job.

Project Plan Part of the RFP involves the development of a **project plan**, which includes an estimate of how long the project will take to complete, an outline of

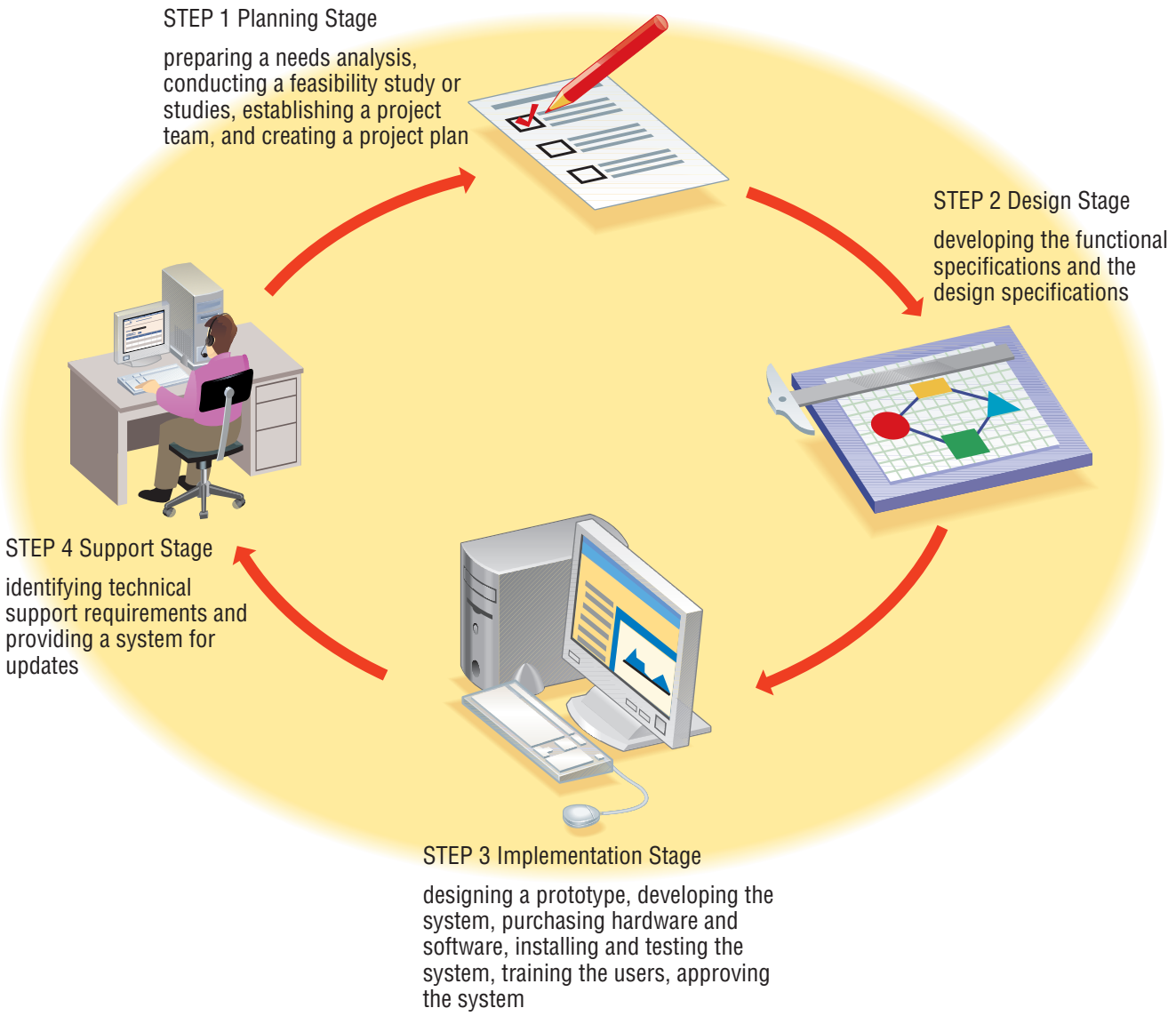

STEP 1 Planning Stage
preparing a needs analysis, conducting a feasibility study or studies, establishing a project team, and creating a project plan
STEP 2 Design Stage
developing the functional specifications and the design specifications
STEP 3 Implementation Stage
designing a prototype, developing the system, purchasing hardware and software, installing and testing the system, training the users, approving the system
STEP 4 Support Stage
identifying technical support requirements and providing a system for updates

Figure 10-5 The System Development Life Cycle
The development of information systems follows an ongoing cycle that includes planning, design, implementation, and support.

the steps involved, and a list of deliverables. A **deliverable** is a document, service, hardware, or software that must be finished and delivered by a certain time and date in order to keep the project on schedule. Big Engines' project team must decide these requirements before the RFP is ready for release. Payments are usually based on the successful completion of project deliverables.

Project planning normally involves a large number of meetings. Systems analysts must meet regularly with potential system users and management personnel to learn exactly what they want and how their demands can be accomplished. Interviews, employee questionnaires, and simple observation of daily practices are all part of this process. Various project management tools are available to help schedule each phase of the project and to determine the priority of each step.

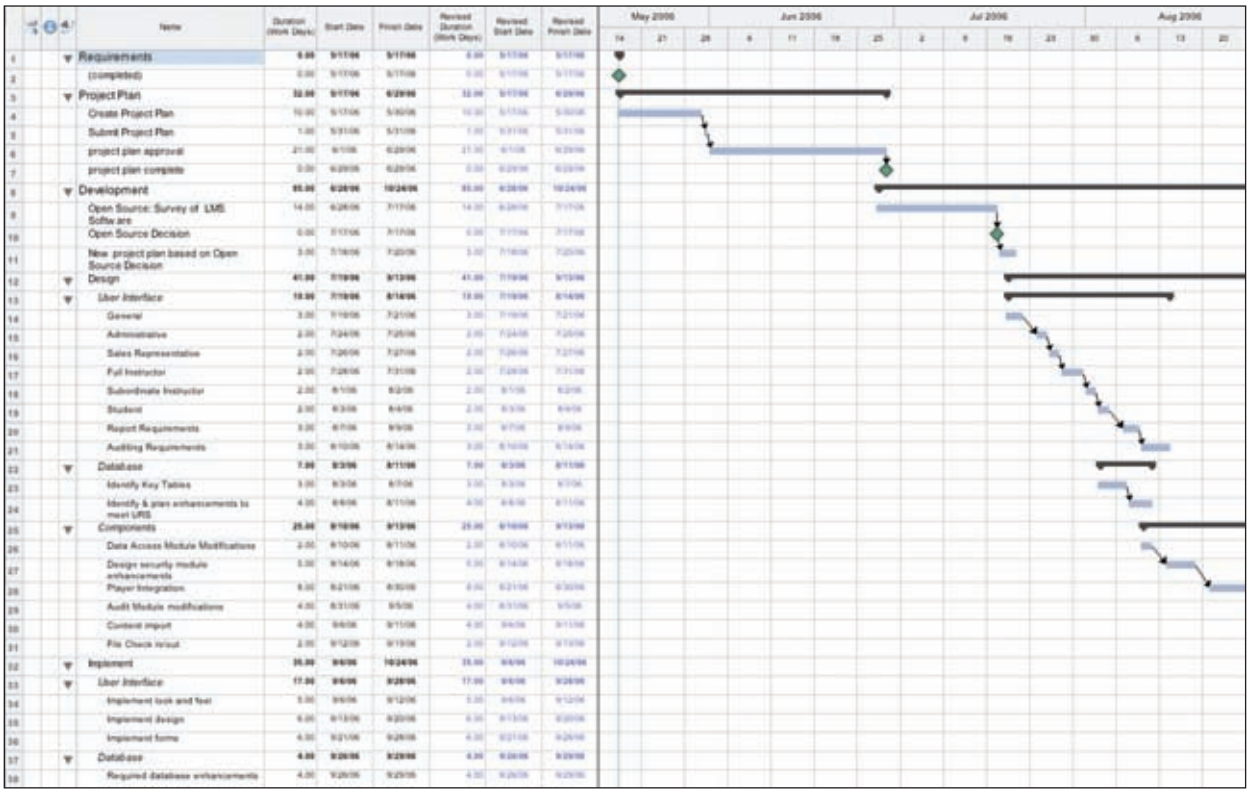
Project Management Tools The project manager would probably use a Gantt chart and computer-aided software engineering (CASE) tools to handle large projects, such

as a CIM system for auto manufacturing. A **Gantt chart**, named after Henry Gantt, is a bar chart showing the stages of a project and the order in which they must be completed. Gantt charts graphically show task beginning and end times and are used for scheduling purposes. Setting realistic time frames is very important to the success of any project and is one of the more difficult tasks of project management.

A CASE tool is any tool used to carry out a project analysis, create a graphic specification, automatically generate and compile program code, test and debug the program(s), provide maintenance, and handle design changes during implementation. They also generate a project database that documents the information system using reports, graphs, models, and flowcharts of processes. This category thus encompasses a large group of sophisticated software packages that automate the design and development of an IS. Several software vendors sell packages that integrate an entire set of CASE tools. Most projects require additional custom programming, however, to closely fit the company's information system needs. Using CASE tools is generally referred to as **rapid application development (RAD)**.

Designing the System

A project is ready to move into the design stage once the project team has approved the plan, including the budget. The design process begins with the writing of the documentation, which covers functional and design specifications. In most cases, the project team creates the functional specification, describing what



Gantt charts help to visually communicate the critical steps and schedule for a project.

the system must be able to do. The company that wins the bid to do the project usually writes a second document, called the design specification. The design specification details the hardware and software necessary to execute the functions. The project leaders must approve both documents before work can proceed.

Tech Demo 10-1 Process Modeling: Entity Relationships

Functional Specification The **functional specification** document is usually written first and states exactly what the information system must accomplish. For example, one part of the functional specification may require that 20 computer workstations be connected to a secure company database, with each workstation having Internet access. This document is concerned only with the functions the system will have to perform, not with the kind of computers or software that will be necessary to operate the system. One part of Big Engines' functional specification might demand that the CIM system automatically send a report back to the orders database every time a car is finished and driven off the factory floor.

Tech Demo 10-2 Process Modeling: Data Flow Diagrams

Design Specification The **design specification** spells out how the goals laid out in the functional specification will be reached. Hardware and software requirements are detailed, including:

Tech Demo 10-3 Process Modeling: Decision Trees

- operating system—Unix, IBM mainframe, etc.
- data model—object-oriented, relational, etc.
- authorized users—discussion of access issues

Part of this step requires the development of a data dictionary listing all the information that must be handled, and the types, names, and sizes these data elements require. As an auto parts company, Big Engines might want an inventory of all raw materials used in production. They would also want records providing ordering information for obtaining new parts from their suppliers. Most important would be their sales orders and customer records.

Another significant part of any design specification is deciding whether to use packaged (off-the-shelf) software or custom software. Big Engines might buy an off-the-shelf Oracle database and a CAM software package such as Wonderware, and then employ a large consulting firm such as Deloitte Consulting or IBM to use those products in the design and development of a custom information system. This software combination would be able to provide a good CIM system covering the entire operation.

Tech Demo 10-4 Process Modeling: Object Modeling

In general, packaged software is cheaper and more reliable, but it may not solve all of a customer's needs. Custom software is more expensive and takes longer to develop, but it can be tailored to the exact needs of a buyer. Table 10-1 lists some commonly used commercial software packages for developing information systems.

Table 10-1 Commonly Used Information Systems Software

Software Type	Commercial Example
database	Oracle Database, Microsoft Access, IBM DB2
financial operations	Peachtree, QuickBooks
CAM/CIM	Invensys Wonderware, FactoryLink

CHAIR, SUPERVISORY BOARD, SAP

Hasso Plattner

EVER HEARD OF BILL GATES and his impact on the world of personal computers? Of course. Ever heard of Hasso Plattner? His name is not nearly as recognizable, but Plattner has had a comparable impact on corporate computing.

Before Plattner came along, software for running a business needed to be custom-made and housed on high-priced mainframes. Plattner cofounded the small company SAP in a small town outside of Heidelberg, Germany, some 30 years ago. SAP has revolutionized the software industry by developing business programs that can be tailored to meet a company's needs and run on its own computers. In an industry ruled by American giants, Plattner's SAP became the world's third-largest software supplier.

Plattner is considered the charismatic visionary of SAP, a man with creative insight into software needs as well as a great marketer and deal-closer. He is a colorful man known for his bitter rivalry with Larry Ellison, the CEO of Oracle, the world's second-largest software company. This competitiveness extends to the yachting world, where the two captains of industry are equally competitive racing their sailboats. Plattner doesn't mind taking on Bill Gates either. Before a large audience at an SAP user conference in 2002, Plattner pleaded with Gates to adopt Java programming language so that software programs from different companies could work together more harmoniously. He used Ronald Reagan's words to Gorbachev, dramatically challenging, "Bill Gates, tear down that wall!"

A multibillionaire by the age of 59, Plattner decided in 2003 to step down as co-CEO of SAP to become chairman of its supervisory board. Granted, recent changes in German law will allow him to be much more involved in the company's activities compared with his counterparts on American supervisory boards. But Plattner says he is eager to be free of the tedium involved in running a company so that he will be able to focus all of his energy on technology and business strategy.

Implementation

The project can move into its next phase, **implementation**, once the development team and the systems house develop the design specification and approve the plans. This step is where the actual work of putting the system together will be done, including creating a prototype and completing the programming. In most cases, implementing the new system is the longest, most difficult step in the process.

Prototype A **prototype** is a miniprogram that demonstrates the user interface and functionality of the system. A prototype may be created if the proposed system represents a major change or is significantly different from existing systems. This step is sometimes skipped, but it is often worth the effort as it allows users to see what the final project will look like, often months before the completion date. Users often generate many suggestions for changes after they review the prototype. These suggestions will be easier to include if they are discovered at this early stage.

Development This is the stage where the software programming is done, and the technicians ensure that the new computer hardware works. This phase involves discovering problems and spending long hours to solve them, and it requires hard work on the part of the development team. The information system will normally use an existing software package, but customization will be required to make everything work as desired. For example, an oil refinery and a grocery store might purchase the same "generic" information system package, but in both cases many modifications would have to be made to make the software meet their needs. A company that can use niche package software precisely tailored to their industry can save considerable time and money.

Hardware and Software Purchasing All of the required hardware and software must be purchased and delivered before any system can be installed and considered up and running. In most cases, the software and service costs far outweigh the hardware costs, as the service requires many hours of highly skilled labor. Integrating all of the system's computers and software and getting the entire system to work as planned involves significant expenditures of financial and human resources.

Installation and Testing Beta testing begins once the computers and software are installed and operating. **Beta testing** is a period during which the system is rigorously tested before it is completely trusted to function as designed. Any previously installed information systems are usually still in operation, and information from the old system can be moved to the new one during this time.

In the Big Engines case, the company would probably choose one of their many factories to test the system. All of the other factories would continue using the company's old system until the new one was deemed satisfactory. Once the new system has been approved, it would be installed in every plant, a process referred to as rolling out the system.

Ordering and installing hardware for a new information system can be an expensive portion of the project budget.

Globetrotting

E-Stonia

ESTONIA IS A SMALL COUNTRY IN THE BALTIC that spent years under the thumb of the Soviet Union. When it gained independence in 1991, it was free to build its own government from the ground up. It chose to develop a style of e-government that streamlines services, reduces paperwork, eases civic involvement, and lets its citizens keep a close watch over government officials.

One percent of the national budget for the next ten years is earmarked specifically for information and communication technology development. Big, blue road signs direct citizens to the more than 700 locations across the country where free Internet access is available. All libraries and schools are also connected to the Internet.

During cabinet meetings, each minister sits in front of a computer screen that can instantly provide any pertinent documents or information. Traveling ministers don't have to miss a meeting—they can participate from any location via Internet. Voting is done online so that decisions appear on the Internet within a minute of taking place.

Citizens can access government Internet sites using their national ID card to vote, file income taxes, access any of their personal records in local or federal databases, and communicate directly with government departments. The government intranet called "X-Road" connects the various information systems across the country.

Citizens can also turn to the "Today I Shall Decide" Web site to propose changes to legislation. If the site registers a significant proportion of citizens lobbying for a proposal, it is sent directly to the appropriate ministry for consideration. In its first year twenty-two citizen-initiated proposals made it into law.

Launching an e-government has been facilitated by the fact that Estonia, with 1.4 million citizens, has a pretty tech-savvy population. Sixty percent of Estonians are connected to the Internet, and 90 percent own cell phones. The country has one of the highest Internet banking rates in the world. And because the Internet provides a window to all government meetings and processes, Estonia is considered one of the least corrupt political systems in Eastern Europe.

User Training User training normally takes place at the same time as beta testing. System users must learn how to run the new software. This process can also result in the discovery of any errors not detected during the design and implementation process. A number of things can go wrong with an information system, including lost records, network errors, system crashes, or the accidental display of confidential information, such as employee salaries. All of these things can cause the development team to be called on to create an emergency patch, or correction, to the system. In the worst cases, errors may cause the team to go back to the planning stage to determine the origin of any problems.

Approval Once an information system is installed and working, the committee or executive overseeing the purchase of the system must inspect, test, and approve it. This is normally done after the system has proved itself during a successful beta

At the approval milestone, the work is officially accepted and final payment is made.

test period. Final approval is often tied to full payment being made to the suppliers. This can be a stressful time for everyone involved if a system has problems.

The Support Stage

A system goes into the support stage after it has been accepted and approved. It is quite possible that undetected errors might still exist, or that additional features may be requested. A **support contract** normally allows users to contact the systems integrator for technical support, training, and sometimes on-site troubleshooting. Even if the system was designed in-house, the responsible department often operates as an independent entity—sometimes even charging the department acquiring the system. The support stage continues until a new information system is proposed and developed, usually years later. At that point, the existing system is retired and no longer used.

Return to Stage One

The system development life cycle is referred to as a cycle because it operates in a circular fashion. Technology moves forward so fast that as soon as one process ends, a new development phase begins. If mistakes have been made the team might have to "go back to the drawing board" and return to the planning stage. This can happen for many reasons, including changes in funding, new technology, errors in the original plan, or other unforeseen difficulties. In any case, the circular methodology helps foster continual refinements and improvements in the process, which in the end benefits all participants.

OnTheHorizon

ALTHOUGH THE EVOLUTION OF INFORMATION SYSTEMS TECHNOLOGY tends to hold a rather steady and straight course with few dramatic breakthrough products, there are nevertheless some new directions in ways to deal with information. Companies are especially interested in new strategies that produce the greatest return on investment, including ways to identify and extract information from existing systems and innovative ways to capitalize on current technology.

Auditable E-Voting Systems

Electronic voting (e-voting) systems are one of the most critical information systems in use in the United States today. Since their introduction, a series of disputed election results have called into question how well these systems can protect against illegal vote manipulation. Compounding this problem is the fact that many e-voting systems in current use cannot be reliably audited because the verification of vote totals relies on the same software used to record the votes. The National Institute of Standards and Technology (NIST) now plans to recommend the use of software-independent e-voting systems that employ alternative audit methods that are less vulnerable to fraud. State-of-the-art systems as envisioned by the NIST are not on the market yet, but will be soon.

Expanded ATM Capability

Automated teller machines (ATMs) are an example of an information system used in everyday life. ATMs seamlessly integrate electronic encryption, identification, database, and financial information systems in order to provide a secure electronic version of the traditional bank teller. However, unlike real-life tellers, ATMs in the United States are unable to accept deposits of checks and cash without the use of an envelope. Having to use an envelope is inconvenient, and adds to the expense of processing deposits. Work is now underway on ATM machines that will be able to accept checks and cash without the use of envelopes, and some machines are already being tested in existing ATM networks. A new federal law known as Check 21 has given a boost to these new machines by legalizing the electronic clearing of checks using scanned digital images, and before too long you can expect to encounter one of these ATMs where you live.

This added convenience is part of a larger trend of extending ATM capabilities. Machines in some countries can already print checks and event tickets, offer foreign currency exchange, and use biometric systems to verify identity. It is only a matter of time before additional technologies transform ATMs into one-stop centers that will allow consumers to conveniently take care of a variety of different transactions and purchases.

Deriving Knowledge from Information

Microsoft founder Bill Gates recently cited management guru Tom Davenport's definition of knowledge as, "information combined with experience, context, interpretation, and reflection," and noted that knowledge derived from information provides a competitive edge. While the Internet has greatly expanded the amount of information available to anyone with a PC and an Internet connection, obtaining knowledge from that information is often difficult, leading Gates to see great opportunities in developing knowledge tools. He views pattern recognition and mental model tools as the next steps in helping people assess the value of the information they have access to. These and other tools will soon facilitate the task of making sense of the information we have at our fingertips.

Chapter Summary

For an interactive version of this summary, go to this text's Internet Resource Center at www.emcp.net/CUT4e. A Spanish version is also available.

What role do information systems play in business?

An information system (IS) is a combination of people, hardware, software, computer networks, and data that organizations use to manage daily and long-term operations. computerized information management increases productivity and efficiency, making companies more competitive. Computer-based information systems offer four key advantages over traditional (pre-computer) information systems: improved speed, efficiency, quality, and measurability.

What functions do information systems perform?

Common information system functional classifications include distribution management systems, office information systems, management information systems, decision support systems, executive support systems, and factory automation and plant operation. Within these categories are a number of different types of information systems, including **office information systems (OIS)**, **electronic data processing (EDP)**, **management information systems (MIS)**, **decision support systems (DSS)**, **online analytical processing (OLAP)**, and **executive support systems (ESS)**. There are two systems used in factory automation, computer-aided manufacturing (CAM) and computer-integrated manufacturing (CIM). **Statistical quality control (SQC) system** is another form of information system often used in plant operations.

What considerations should be made when planning or updating an information system?

Upgrading or switching to a computer-based information system affects technical staff, information technology costs, profit margins, product quality, and clerical staff. Companies should also realize that there will be disruptions. Once the impact of a new system has been studied and a decision to proceed has been made, a new system must be purchased "off the shelf," or developed and brought into service. **Outsourcing** simply means hiring a third party to handle a project. A **systems integrator** is a company that specializes in installing and supporting information systems.

What are the important factors to consider when choosing a new information system?

The following factors should be considered when choosing an information systems package: user interface design, **open system** versus **closed system, niche information system**, **turnkey system**, and system support.

How is a new information system developed?

Because of their large size, information systems require the creation of a project team. Sometimes the project manager also functions as a **business systems analyst**, responsible for completing the systems analysis and making design recommendations. Creating a development plan and executing it has four main stages: planning, design, implementation, and support. The **system development life cycle (SDLC)** process is a circular process, meaning that once the last stage is reached the planners return to the beginning and repeat the process. A **feasibility study** is conducted, investigating how difficult the project might be to complete and how much it might cost. If management decides to hire a systems house, a **request for proposal (RFP)** will be sent to one or more possible suppliers. In most cases, the project team creates a functional specification, describing what the system must be able to do. **Design specifications** detail the hardware and software necessary to execute the functions. A **prototype** is a miniprogram that demonstrates the user interface and functionality of the system. **Beta testing** is a period during which the system is rigorously tested before it is completely trusted to function as designed. User training normally takes place at

the same time as beta testing. All of these things can cause the development team to be called on to create an emergency patch, or correction, to the system. A system goes into the support stage after it has been accepted and approved. A support contract normally allows users to contact the systems house for technical support, training, and sometimes on-site troubleshooting.

KeyTerms

Numbers indicate the pages where terms are first cited in the chapter. An alphabetized list of key terms with definitions (in English and Spanish) can be found on the Encore CD that accompanies this book. In addition, these terms and definitions are included in the end-of-book glossary.

Information Systems Classified by Function

distribution management system, 445
just-in-time (JIT) distribution, 447
stockless inventory distribution, 447
office information system (OIS), 448
standard operating procedure (SOP), 448
connectivity, 448
electronic data processing (EDP), 449
management information system (MIS), 449
decision support system (DSS), 450
data modeling, 450
expert system, 452
knowledge base, 452
online analytical processing (OLAP) system, 453
business intelligence (BI), 453
executive support system (ESS), 453
computer-aided manufacturing (CAM), 453
computer-integrated manufacturing (CIM), 453
statistical quality control (SQC) system, 454

Considerations for Planning and Updating Information Systems

outsourcing, 458
systems integrator, 458

Factors to Consider When Choosing a New Information System

open system, 459
closed system, 459
niche information system (vertical market package), 459
turnkey system, 459

The Process of Developing a New Information System

business systems analyst, 460
software engineering, 461
system development life cycle (SDLC), 461
feasibility study, 462
request for proposal (RFP), 462
project plan, 462
deliverable, 463
Gantt chart, 464
rapid application development (RAD), 464
functional specification, 465
design specification, 465
implementation, 466
prototype, 466
beta testing, 467
support contract, 469

ChapterExercises

The following chapter exercises, along with new activities and information, are also offered in the Internet Resource Center for this title at www.emcp.net/CUT4e.

Tutorial > Exploring Windows

Tutorial 10 teaches the steps for installing and removing software so that the Windows Registry correctly reflects the updates to its database.

Expanding Your Knowledge > Articles and Activities

Visit the Internet Resource Center for this title at www.emcp.net/CUT4e, read the articles related to this chapter, and complete the corresponding activities. The article titles include:

- Topic 10-1: Value-Added Resellers and Original Equipment Manufacturers
- Topic 10-2: Horizontal vs. Vertical Market Software
- Topic 10-3: Support Contracts
- Topic 10-4: The Buy vs. Build Decision

Terms Check > Matching

For additional practice, go to the Internet Resource Center for this title at www.emcp.net/CUT4e for a chapter crossword puzzle.

Write the letter of the correct answer on the line before each numbered item.

a. functional specification
b. prototype
c. Gantt chart
d. beta testing
e. turnkey system
f. outsourcing
g. stockless inventory
h. user training
i. design specification
j. decision support systems (DSS)

____ 1. The process performed to ensure that a new information system is working properly.

____ 2. The document describing the software and hardware needed for any new information system.

____ 3. The practice of hiring technical consultants from other companies to help with information systems development.

____ 4. A graphic depiction of the steps in an information system development schedule.

____ 5. A procedure required with any new information system so that people can learn how to do their jobs.

____ 6. Systems built to perform data modeling and provide "what-if" analysis.

____ 7. An early step in information systems development, which allows people to see in advance what the final system will look like and how it will operate.

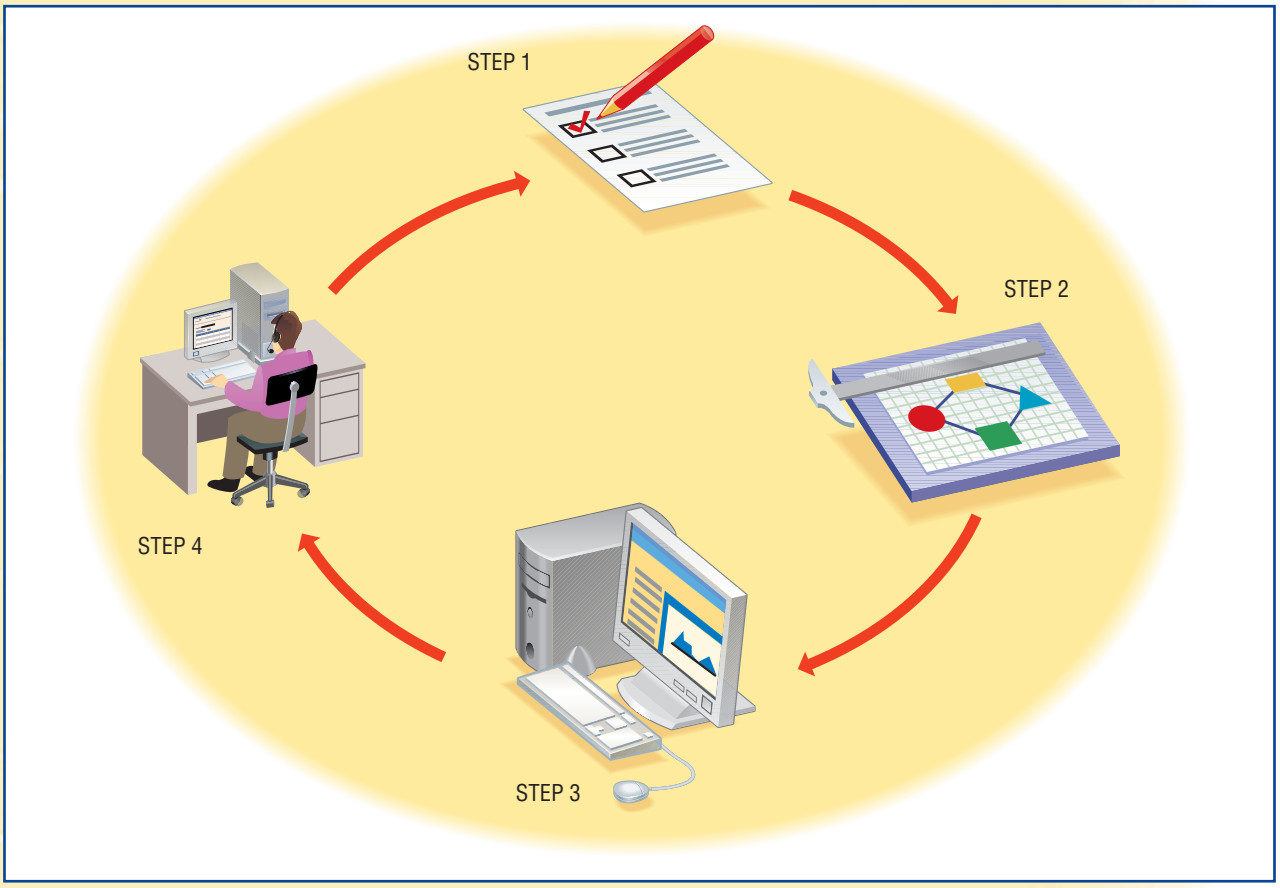
____ 8. An expensive custom system that is easy to use and performs every function the user desires.

____ 9. A document describing how a new information system will operate.

____ 10. A new distribution strategy that uses an information system to maintain inventories with daily deliveries of product.

Technology Illustrated > Identify the Process

What process is illustrated in this drawing? Identify the process and write a paragraph describing its use.



Knowledge Check > Multiple Choice

Additional quiz questions are available on the Encore CD that accompanies this book as well as on the Internet Resource Center for this title at www.emcp.net/CUT4e.

Circle the letter of the best answer from those provided.

1. A type of information system with in-depth knowledge about a specific subject is called a(n) _____ system.
   a. integrated information
   b. strategic information
   c. executive information
   d. expert

2. This type of information system is used to track measurements of final product quality.
   a. quality control system (QCS)
   b. process control system (PCS)
   c. strategic information system (SIS)
   d. statistical quality control (SQC)

3. A system that allows purchasers to make their own configuration changes is called a(n)
   a. closed system.
   b. open system.
   c. freeware.
   d. prototype.

4. Which of the following are software applications that help in rapid application development?
   a. CASE tools
   b. spreadsheets
   c. design specifications
   d. beta test operating systems

5. Which of the following steps would normally occur first when developing a new information system?
   a. Writing the design specification
   b. Creating a prototype
   c. Interviewing people to discover what they want
   d. Beta testing

6. Which of the following documents describes what a system needs to do, but not how it should be accomplished?
   a. prototype
   b. functional specification
   c. design specification
   d. user manual

7. Which of the following describes a system for providing supplies to a factory at the last moment before production?
   a. just-in-time delivery
   b. stockless inventory
   c. batch processing
   d. transactional processing

8. This type of information system does not allow users to modify the software.
   a. closed system
   b. open system
   c. niche information system
   d. shareware

9. A company that specializes in installing and supporting information systems is called a
   a. distribution management system.
   b. system integrator.
   c. system development life cycle.
   d. decision support system.

10. Spreadsheets are often used for modeling data in a process called
    a. running the numbers.
    b. expert systems.
    c. data mining.
    d. batch processing.

Things That Think > Brainstorming New Uses

In groups or individually, contemplate the following questions and develop as many answers as you can.

1. One of the main benefits of system prototyping is that it allows users to see how the system will work long before development and testing is completed. Not only does this allow the project team to ensure that the system will meet the users' requirements, but it also makes the users feel more comfortable about the project. In what other industries would prototyping a product make sense?

2. Expert systems have been developed to determine the appropriate antibiotic for physicians to prescribe, to forecast the avalanche hazard for a particular area, and to determine whether to approve a credit application. Expert systems are particularly useful when determining whether the outcome may be costly or dangerous. Think of three other processes or decisions that would be made easier through the use of expert systems.

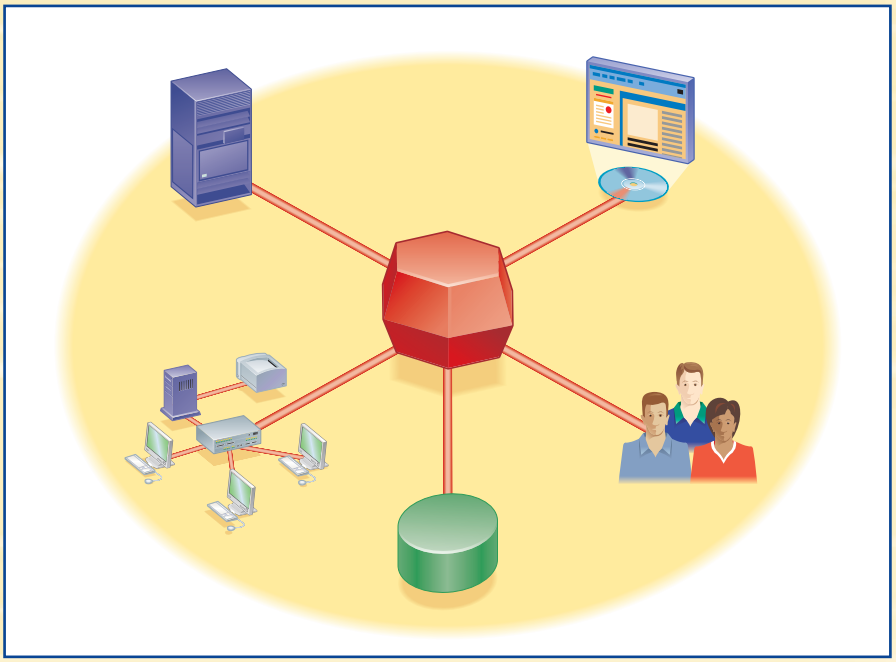
Key Principles > Completion

Fill in the blanks with the appropriate words or phrases.

1. Systems that are heavily customized to meet the exact needs of a business are called _____.
2. A(n) _____ demonstrates the user interface and functionality of the system to allow users to see what the final project will look like.
3. The ability to link with other programs and devices is called _____.
4. A set of facts and rules called a(n) _____ is a key componet of an expert system.
5. Because it must be serviced and supported by the original vendor, a(n) _____ often costs more money than expected.
6. The _____ is a series of steps culminating in a completed information system.
7. _____ are documents, services, hardware, and software that must be finished and delivered by a certain time and date in order to keep the project on schedule.
8. Systems that help managers make informed business judgments are called _____.
9. _____ can be modified by a company's IT staff without involving the original vendor of the system.
10. _____ systems use a combination of data tracking and data modeling to build a picture of how well a factory is operating.

Tech Architecture > Label the Drawing

In this illustration of an information system, identify the five components.



Techno Literacy > Research and Writing

Develop appropriate written responses based on your research for each item.

1. How many information systems are tracking your actions right now? Research the different kinds of records stored on government and corporate information systems. What might the IRS, police, and government officials know about you from records in their information systems? Are your school records computerized? What type of information about you is kept by your school? Who is allowed to see your records? Are you allowed to see them?
2. Is there a company in your community that uses a noncomputerized information system? Find an information system that has yet to be converted to a computerized system. Describe the system and how it works. Find out from the system owners why they haven't moved to a computerized system and whether they have plans to do so in the future.
3. How was it done in the past? Conduct research using the Internet and learning resource centers to find out how large information systems worked before the era of computerization. What kind of systems did large corporations use in the 1930s? Describe some of the methods they used and what kind of technology was employed.

Technology Issues > Team Problem-Solving

In groups, brainstorm possible solutions to the issues presented.

1. You have been given the task of preparing a request for proposal (RFP) to send out to multiple systems integrators for the development of a new information system. In order for your company to select the best systems house for the job, you will need to gather information about how each systems house will complete the project. What questions do you need to ask in the RFP to gather information about each systems integrator's project plan and deliverables? Is the timeline for completing the project or the price more important? What other factors do you need to consider before selecting the right systems integrator for the job?
2. Your class has decided to create an information system to track student contact information and assignment due dates for a large group project. Thinking about the steps in the system development life cycle (SDLC), outline the process you would follow to develop the information system. How should the class determine what information should be stored in the system? How will the class agree that the system meets its objectives?

Mining Data > Internet Research and Reporting

Conduct Internet searches to find the information described in the activities below. Write a brief report summarizing your research results. Be sure to document your sources using the MLA format (see Chapter 1, page 42, to review MLA style guidelines).

1. An information system operates on your school campus. Try to find out how that information system is organized. Can department assistants access any type of student information? Could they change data from their desks, using the right password? Is class schedule information kept separately from grades, parking tickets, library fines, and other types of information? How much of a student's information is available over the Web?

2. Imagine you are operating your own used-DVD business using an information system running on a single computer. What features would you want it to have and what equipment would you want it to run on? How could you use this system to expand your business capabilities and increase sales? What portions of the information system would be useful to have accessible on the Web?

3. Learn how an information system works at a local store by observing and asking questions during a visit. Does the system track inventory? Does it print receipts, give lists of new products to order, and allow for returns? What other features does it have? Is the owner happy with the system? Are there any other features the owner would like the system to have?

Technology Timeline > Predicting Next Steps

Personal communications technology is growing pervasive, making it difficult to "hide" from family, friends, or employers. This trend is likely to continue as high-tech electronic gadgets bring us all into constant contact. Look at the timeline below outlining the major benchmarks in personal communications technology. Think of what might happen next as our communications capabilities expand. Add events to this timeline up to 2015.

1980 The first widely available cellular phone systems are offered to the public.

1993 The Internet takes off, and e-mail messages replace letters as a popular form of communication.

2000 Child location devices are introduced. Young children can call for help by pushing a button on their backpacks, and warnings can be sent if they leave their school area.

2001 E-mail messages sent via cell phones become commonplace.

2002 Cell phones with digital cameras are available in the United States.

2003 Video instant messaging (VIM) is introduced, combining video and audio with instant text messaging.

2009 Google Latitude is launched, allowing friends and family to track each other's physical location via a cell phone or PC.

Ethical Dilemmas > Group Discussion and Debate

As a class or within an assigned group, discuss the following ethical dilemma.

When a company implements a new information system, some of management's goals for the project may include improved quality, increased profits, and reduced staff. The participation of current staff in the project is needed to ensure that the new system supports the business properly, but sometimes employees are reluctant to support the project for fear that their jobs are at stake. If staff is resistant to the changes that the new system will bring, the company may not successfully complete the project and realize its goals. Is it fair for management to expect staff to cooperate with the project if they may lose their jobs as a result of the successful implementation of the new system? Is it fair for staff to delay or sabotage the project for the sake of job security? What alternatives might management consider in order to keep employee morale high and complete the project successfully?

CHAPTER 12

Programming Concepts and Languages

Learning Objectives

- Define programming and describe the four classic programming elements
- Distinguish between low-level and high-level programming languages and discuss the different language generations
- Describe the divide-and-conquer approach and identify the standard problem-solving steps
- Explain the common types of programming errors
- Discuss the different phases of the software development life cycle
- Identify the major programming languages and discuss their features

CyberScenario

JAMES GOT THE CALL EARLY TUESDAY MORNING. A pulp and paper plant in the Cascade Mountains had experienced a system failure, and they were in a "downtime" situation that meant their plant's operations were not running at all. They wanted James's help because he had written the plant's control system software. James asked numerous questions in an attempt to locate the source of the trouble.

While he was still speaking on the phone, he used his computer to log in to the plant's network, but the nature of the failure prevented him from making a remote diagnosis.

James told the company's IT staff that he could not fix the problem from his office. He knew that they would ask him to honor the support contract he had signed and fly to Oregon immediately. There was a short discussion at the other end of the phone, and James was told that the company would offer him $2,000 a day plus expenses. As an added incentive, he would receive a handsome bonus if he fixed the problem within one day. The system failure was already costing them thousands of dollars an hour, so any money spent on repairing the problem was miniscule compared with the money they were losing. James agreed to come right away. He hung up the phone, packed his bags, and raced for the airport. On the way

he pondered his life as a software programming consultant—long spells of boredom punctuated by brief periods of excitement when everybody wanted him at once.

By late afternoon James was driving a rental car up winding mountain roads. As he neared the plant, he winced at the acrid pulp-mill odor. After a brief chat with the IT personnel, he logged in as a system administrator to look at the system log files to find the problem. James carefully narrowed down the possible sources of trouble by working step by step through logged events, and by questioning the operating staff. In a few hours he found a defective programmable logic controller (PLC) module and replaced it. The problem was repaired, and the plant went back into full operation. As was often the case, the customer had blamed software for a hardware failure. Fortunately, James was experienced enough to check all of the possibilities. James left behind an invoice for his services and caught the last flight out of Portland.

James's experience illustrates the key role that software programmers play in supporting our present-day lifestyle. Without programmers, factories would grind to a halt, communications systems would shut down, and many of the modern conveniences that we take for granted would be unavailable.

Programming Concepts

A **program** is a set of instructions telling a computer how to perform various tasks, and programming is the act of creating these instructions. Programmers use a **programming language** to create a program. Programs are also referred to as **source code**. **Coding** is a term programmers use to refer to the act of writing source code.

Large programming projects are much like engineering projects. They are too large for a single individual to complete, so they require a programming team. For example, a professionally released, mass-market software program may be the result of the combined efforts of five programmers, or of more than five hundred. Each programmer in a programming team is responsible for a small portion of the final program, and team leaders must carefully coordinate the efforts of the team members to ensure that the program works as intended. Software engineers typically coordinate programming projects while working with programming teams.

INTERNET RESOURCE center

Go to this title's Internet Resource Center and read the article titled "Logic Tables and Control Structure." www.emcp.net/CUT3e

Language Characteristics and Classifications

Programming languages contain smaller vocabularies than human (natural) languages and are much less complex. A typical programming language may contain 100 to 200 elements (words and symbols), whereas a human language such as English contains around ten thousand elements. Programming language **syntax**, or structure, also tends to be less complex than human language syntax. A programming language is therefore not as difficult to learn as a foreign language, such as French or German.

The process of writing a program is similar to the process of writing a term paper. A writer chooses a topic and then thinks about how the topic can be organized in a logical fashion to ensure that the paper will convey the writer's intent. During the writing process, a term paper goes through several drafts, which are corrected and updated until a final version is ready. Programming follows the same steps, but uses a programming language and computers instead of a human language and word processing software.

Contrary to popular belief, computer programmers are not typists with a knowledge of computers. Although typing is generally a required skill, many programmers do not type rapidly, even after years of working on computers. They actually spend much more time staring at computer screens and printouts, trying to solve a problem. Successful programmers are measured by their programming skills, not their typing speed, just as chessmasters are measured by the thoughtfulness of their moves, not the speed by which they slide pieces over the board.

Programming computers is analogous to the slow, careful, and mental challenge of playing chess.

Programming is difficult mental work, not unlike the effort involved in writing a report for a college course. Programming is actually more difficult than writing a report, because a program must accomplish all of its goals with zero or little tolerance for error. A professor may gloss over a few minor punctuation mistakes and still give a paper an A, but a program with a single **syntax error** (a mistake in the way programming elements are strung together) will not work at all, and is therefore worthless. This is the equivalent of a term paper being unreadable because of a single misplaced comma.

Go to this title's Internet Resource Center and read the article titled "Linear Programming and Simplex Method." www.emcp.net/CUT3e

High-Level vs. Low-Level Languages Computer languages can be classified as being low level or high level. A **low-level language** is closer in form to the thought processes computers use. Also called **machine code**, this type of programming language is a binary language consisting of 1s and 0s. On the other hand, a **high-level language** is relatively similar to natural languages such as English, making this language type easier to use than machine code. Although high-level languages are easier to learn and use, low-level languages run faster and take up less disk space. This means that software companies can save time and money using high-level languages to develop software programs. However, the programs will take up more space and run more slowly than programs developed using a low-level language.

Programming Language Generations A computer **programming language generation** refers to a group or type of programming languages that were developed at the same time, in a particular chronological order. The language of the most recent generation builds on the languages in the preceding generation. Currently there are five generations of computer programming languages, as defined in Figure 12-1. A computer language is sometimes referred to as belonging to a generation of languages. For example, a programmer might describe HTML as a popular fourth-generation language (4GL) used to design Web pages.

Classic Programming Elements

All computer programs contain four main programming elements, which work together as discrete units or in combined form. Together these program elements allow computers to process data and turn it into information for various uses. The four main programming elements are:

- variables
- executable statements
- looping
- decision statements

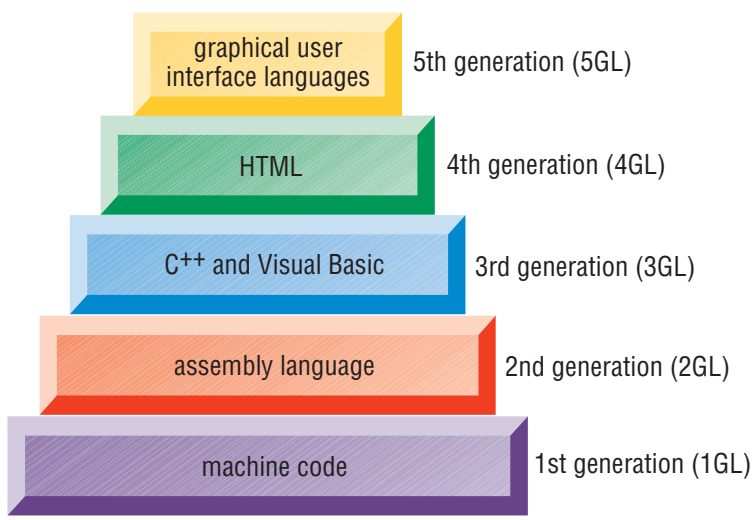


Figure 12-1 Programming Language Generations
Each generation of programs builds on the contributions of the group of languages that preceded it.

Variables A **variable** is a data value stored in computer memory. Variables can be anything a programmer wants to keep and work with—names, numbers, or even a status indicator such as yes/no. For example, a program for adding two numbers and producing a sum would have variables for the first and second numbers, and a variable for the sum of those numbers. Variables normally have data types, similar to the possible data types fields use in a database.

Clicking the enter key may direct a program to perform an executable statement, which will allow the program to proceed to the next statement in a programming sequence.

Executable Statements Statements are used to manipulate variables or for performing actions such as sending output to a screen or to a printer. There are also statements that read input from the mouse and keyboard. If the data is valuable, the program will store that data for later use as a variable. When run by a program, an **executable statement** performs an action and then proceeds to the next statement in the sequence.

Looping An essential element of programming is the ability to repeat parts of a program. A program for controlling a stoplight clearly demonstrates this capacity as it repeats the same sequence of actions and decisions endlessly. **Looping** allows a program to return to a previously executed instruction and repeat it (Figure 12-2). The power of looping is perhaps the single greatest capacity of the computer, as

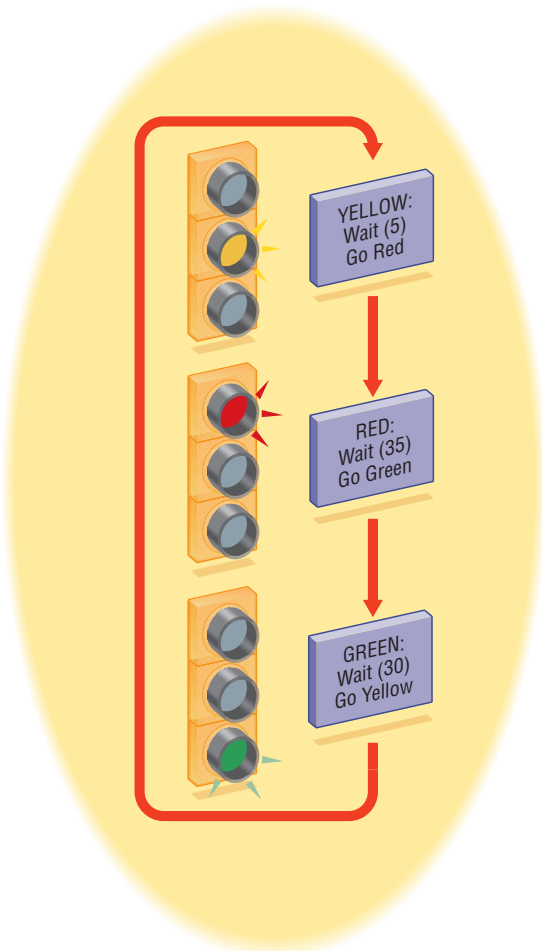
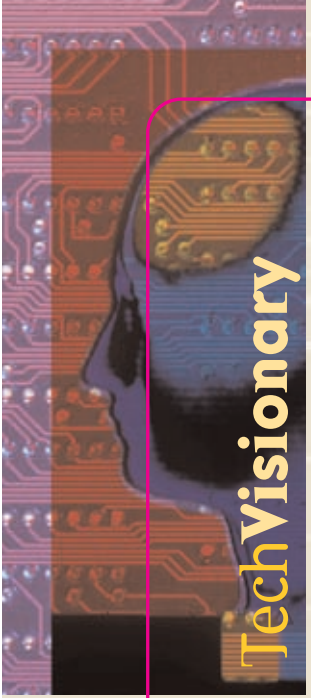


Figure 12-2 Looping
Looping allows a computer program to continuously repeat the same steps, such as a program designed to direct a traffic light to display yellow, red, and green lights at a consistent rate.



THE FIRST PROGRAMMER

Lady Lovelace

DAUGHTER OF THE POET LORD BYRON, Augusta Ada, countess of Lovelace (1815–1852), became Charles Babbage's most enthusiastic supporter. She wrote programs, or sequences of instructions, for Babbage's invention, the Analytical Engine and made several innovations that are central to programming today. These included the subroutine, the loop, and the conditional jump.

THE SUBROUTINE A subroutine is a part of a program that can be used repeatedly. An example of a subroutine is a sequence of instructions that sorts numbers into ascending order.

THE LOOP A loop is a sequence of repeating instructions. A loop works by means of a command that tells the computer to back up to a previous command, as in the following sequence, which loops endlessly, printing numbers from 0 to infinity by increments of 0.5:

```
10  LET N = 0
20  PRINT N
30  LET N = N + .5
40  GO TO 20
```

THE CONDITIONAL JUMP A conditional jump is an instruction that tells the computer to go to a different instruction if a certain condition is met. The conditional jump has many uses in programming. For example, it can be used to keep a loop from repeating endlessly, as in the following set of instructions, which prints numbers from 0 to 6.5 by increments of 0.5:

```
10  LET N = 0
20  PRINT N
30  LET N = N + .5
40  IF N ≤ 6.5 GO TO 20
50  END
```

Source: Shepherd, Robert D. *Introduction to Computers and Technology*, Paradigm Publishing: October 1997.

repetition is the dullest part of any mental activity. Humans naturally become bored when required to repeat a series of actions over and over again. This is not a problem for computers because they never get bored with performing even the simplest and most repetitive chores. Unlike humans, computers excel at doing the same thing again and again very quickly.

Decision Statements A program that simply follows a linear sequence of actions without variation is of limited use. However, if a program consists of a

decision statement, a point in a program where different actions may be performed depending on specific conditions, the program can be more useful. For example, the stoplight program show in Figure 12-2 can be more helpful to pedestrians if the pattern can be interrupted when a pedestrian wants to cross the street. As illustrated in Figure 12-3, the program can be designed so that a crosswalk button controls the stoplight behavior using an **if-then statement**. This statement allows the program to perform differently depending on whether or not the button has been pressed. Computers make decisions by comparing values in variables and then performing different sequences of actions depending on the outcome.

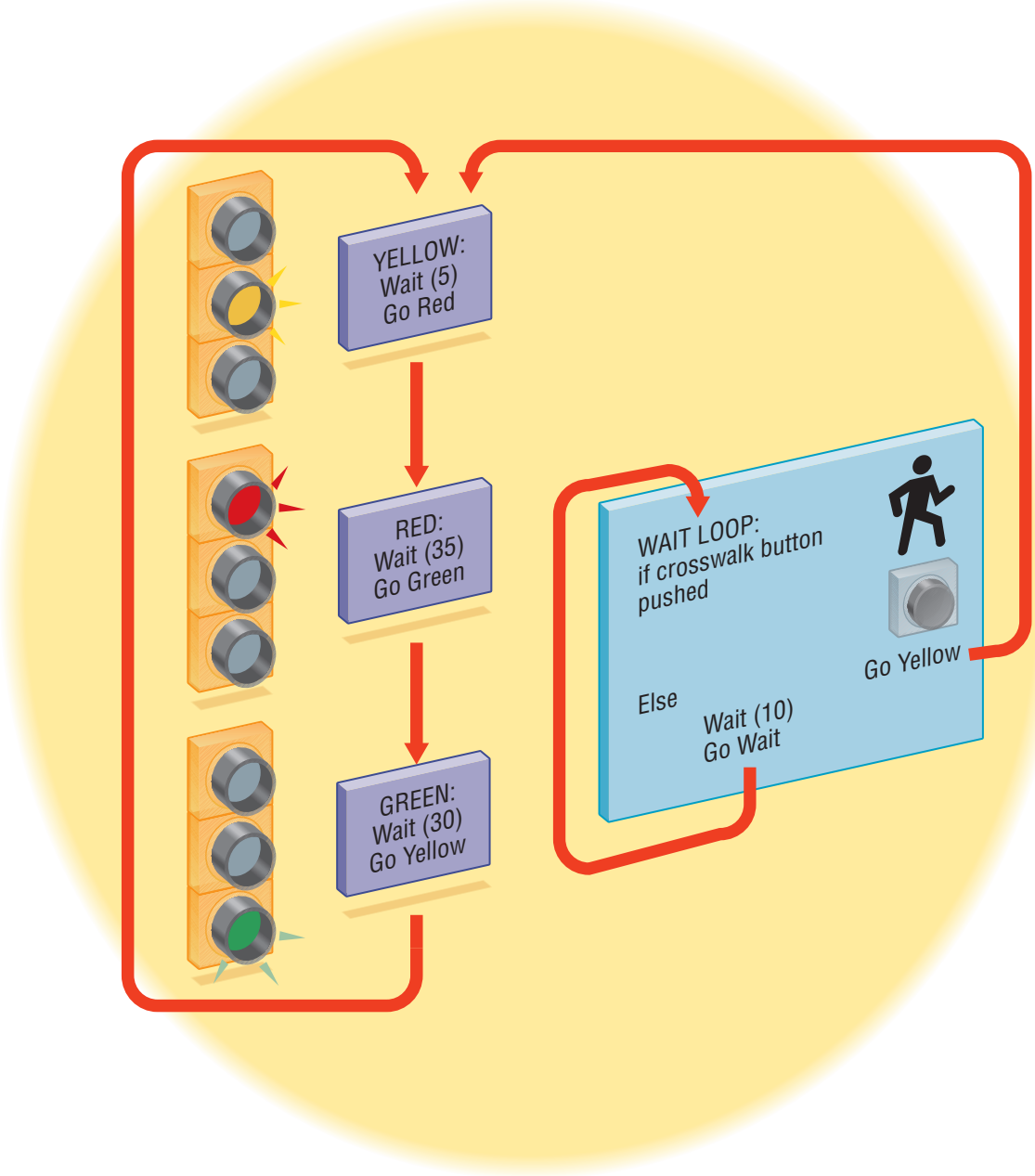


Figure 12-3 A Decision Statement's Effect on Program Looping
Using an if-then statement, based on a particular action, such as pushing the crosswalk button, a program can interrupt the looping pattern, making the program more useful.

Problem-Solving Techniques

Programming is a **problem-solving process**. For programmers this means that technical knowledge alone is not enough. In order to be truly successful they must be creative analytical thinkers. In many cases the technical details of a project are too complex for managers to deal with, so employers prize programmers with excellent problem-solving capabilities. Problem solving is as much an art as a science, but there are techniques for solving problems using logical processes, such as the divide-and-conquer approach and step-based problem solving.

Divide-and-Conquer Approach

One technique to solve problems when developing software is to break down the larger problem into several small problems. It is based on the idea that every program is like a giant puzzle. In the **divide-and-conquer approach**, programmers tackle one small piece of the puzzle at a time. They work on each piece until it is solved, and then move on to the next one.

Before breaking a large project down into pieces, a programmer must develop a strategy for tackling the work. The idea is to envision the project in its entirety in order to view the larger elements that must be handled first. This process is repeated until the entire project is reduced to a long series of small steps, each involving smaller and smaller elements. The entire process can be documented

Before beginning programming work, a programmer must be able to communicate with the company's business leaders to understand the scope and needs of a project.

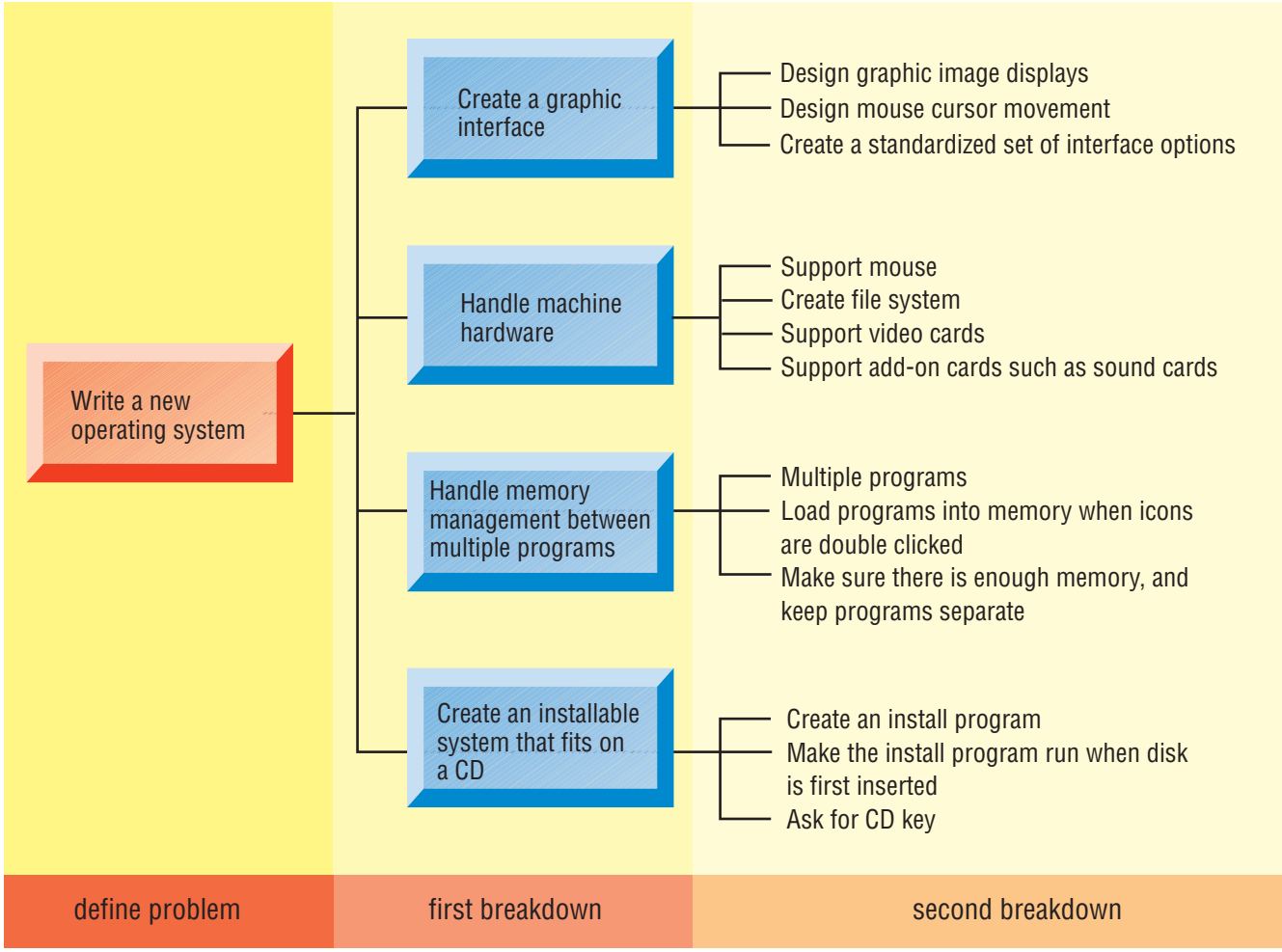

Write a new operating system
Create a graphic interface
Design graphic image displays
Design mouse cursor movement
Create a standardized set of interface options
Handle machine hardware
Support mouse
Create file system
Support video cards
Support add-on cards such as sound cards
Handle memory management between multiple programs
Multiple programs
Load programs into memory when icons are double clicked
Make sure there is enough memory, and keep programs separate
Create an installable system that fits on a CD
Create an install program
Make the install program run when disk is first inserted
Ask for CD key
define problem
first breakdown
second breakdown

Figure 12-4 Example of the Divide-and-Conquer Approach

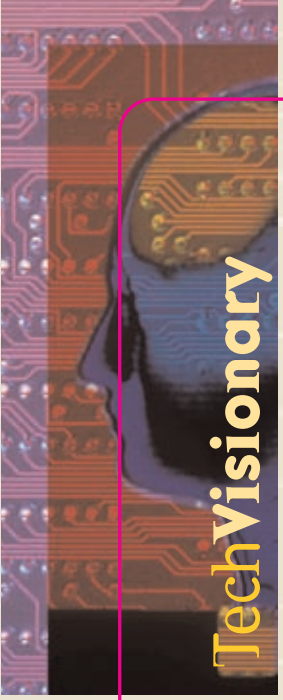
The top-down design approach helps programmers break a large project into manageable parts. The breakdown process continues until there are no more steps.

using an outline format. It is often called **top-down design** because programmers start at the top and work their way down to the bottom. Figure 12-4 illustrates an example of this process.

Problem-Solving Steps

There are few jobs more complex than computer programming. Because of the difficulty, programmers need to be the type of workers referred to as "self-starters." They have to be prepared to take on programming projects with only vague goals and guidelines to produce a working piece of software. To accomplish their tasks they follow a standard set of problem-solving steps:

1. Identify the problem.
2. Analyze the problem.
3. Brainstorm solutions and choose the best one.
4. Write the algorithm.
5. Prototype the solution.
6. Implement and test the solution.

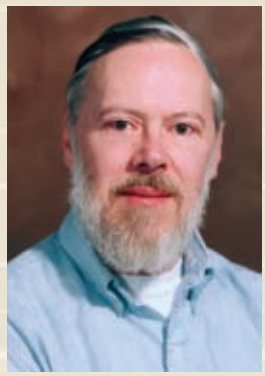


INVENTOR OF THE C PROGRAMMING LANGUAGE

Dennis M. Ritchie

IT MAY NOT BE SURPRISING that great things were in store for the young doctoral candidate whose thesis was on the subrecursive hierarchies of functions. But despite mastering a subject completely arcane to most of us, Dennis M. Ritchie maintains that his college years taught him that he was not smart enough to be a physicist or a mathematician. It did, however, spark his interest in computers and procedural languages.

After finishing his PhD in mathematics at Harvard in 1967, Ritchie followed his father's footsteps to Bell Laboratories. There he met and began working with Ken Thompson and others on the Multics project, with the goal of designing a general computer operating system. From this project, the Unix program, written in assembly language, evolved in 1969. At Bell Labs, Ritchie also experimented with Thompson's B language, adding new data types and syntax to come up with the C programming language in 1972. Soon thereafter, Unix was rewritten in C to capitalize on the language's power and flexibility. C's use of modularity enabled Unix to be portable and to execute quickly. C has had a wide range of applications, including in computer games.



In 1988 Ritchie was elected to the U.S. National Academy of Engineering, and in 1998 Dennis Ritchie and Kenneth Thompson were awarded the National Medal of Technology for their work on the Unix system. Today Ritchie continues at Bell Labs as the manager of a small research group working to take operating systems, languages, and routing/ switching hardware to the next level.

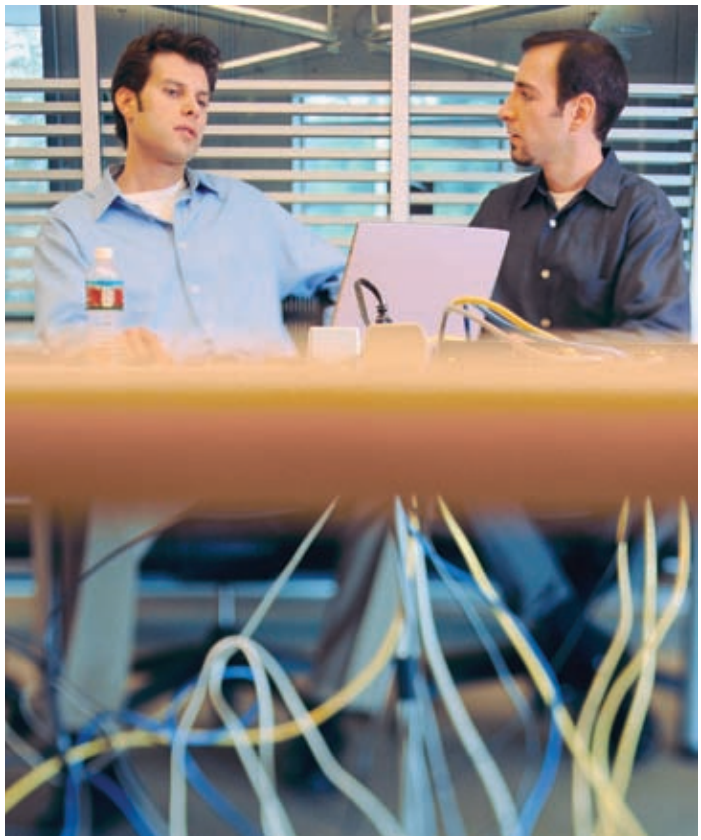
Sources: Home page, *Dennis M. Ritchie*. December 2006 <http://www.cs.bell-labs.com/who/dmr>; "Dennis M. Ritchie," *Lucent Technologies*, <http://www.bell-labs.com/history/unix/ritchiebio.html>.

Identify the Problem The organizations contracting a programmer's services are not always sure what they want. Sometimes they may be able to describe their needs, but not in a way that would help the programmer know where to begin. Programmers may hear statements such as, "Our billing system is too slow; fix it," or, "We would like to network all of our company offices."

Programmers must first hammer out the definition of the problem before they can begin to solve it. A programmer starts every project like an interrogator, beginning with a thorough set of questions covering every aspect that is not clear. If programmers begin a project without correctly identifying the problem(s) that must be solved, they are very likely to come up with the wrong solutions. This means they end up losing valuable time, and quite possibly their jobs. In the case of a slow billing system, a programmer might start off with questions isolating the cause of the slowdown. Is the network running slowly? Are the computers old and out of date? Is there faulty hardware somewhere? A simple fix such as new network equipment or a printer will often solve problems and avoid the need for expensive programming.

Analyze the Problem Once a problem has been identified, the next step involves analyzing it. At first glance, this may seem identical to the first step (identify the problem), but it is more complex. Analyzing the problem can involve learning new skills or concepts. For example, if a problem has been identified that requires the installation of a network connecting 100 computers, the programmer must understand how networks function, what kind of software and hardware are needed, and how it all works. If a solution to an identified problem requires writing a program in Visual Basic, then part of understanding the problem includes understanding Visual Basic. At this point, it is probably clear that the term *programmer* only partly covers what a computer specialist does. How broad a programmer's job becomes is partly related to the size of the organization that the programmer works for. In some jobs, hundreds, or even thousands, of computer people work on a project and in such cases a given programmer's job is very focused. It is more common, however, to have a single programmer or a small group of programmers working on a given project. This widens the list of responsibilities for the individual.

Brainstorm Solutions and Choose the Best One Once a problem has been identified, analyzed, and understood, programmers are ready to develop a set of solutions. Novice programmers sometimes ignore this critical phase. It may appear that the easiest approach would be to choose the first solution that comes to mind, but unfortunately this often leads to less than optimal outcomes. The best strategy is to create a list of all possible solutions, evaluate them, and then choose the best one before proceeding.



By brainstorming, programmers can identify all possible solutions first and then evaluate them to determine the best solution.

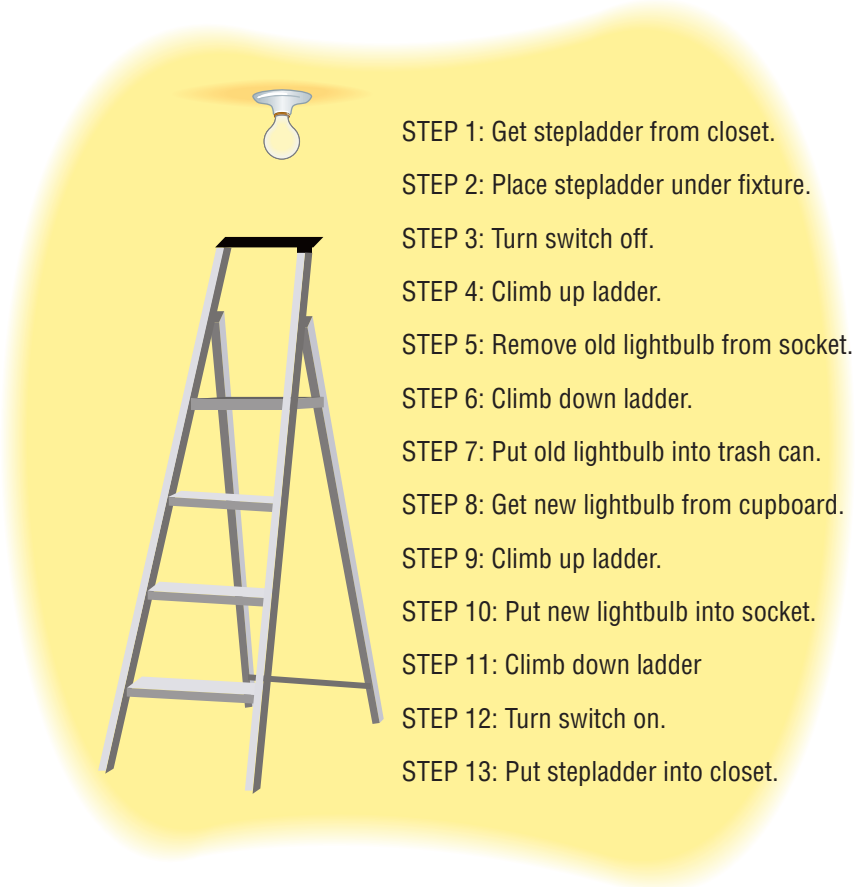

STEP 1: Get stepladder from closet.
STEP 2: Place stepladder under fixture.
STEP 3: Turn switch off.
STEP 4: Climb up ladder.
STEP 5: Remove old lightbulb from socket.
STEP 6: Climb down ladder.
STEP 7: Put old lightbulb into trash can.
STEP 8: Get new lightbulb from cupboard.
STEP 9: Climb up ladder.
STEP 10: Put new lightbulb into socket.
STEP 11: Climb down ladder
STEP 12: Turn switch on.
STEP 13: Put stepladder into closet.

Figure 12-5 Step-by-Step Pseudocode Algorithm for Changing a Lightbulb

Write the Algorithm The next step involves clearly defining the solution. If it is determined that new software is needed to solve the problem, part of this step may include deciding which language to use in creating the program. To assist in defining the solution, programs are first written out as an algorithm before being written in a programming language. An **algorithm** is a complete list of steps for solving a problem. Algorithms usually are written in **pseudocode**, which is a very high-level language that computers cannot read. Pseudocode is a useful tool for organizing the way a programmer thinks about a solution. The algorithm is later translated into a program the computer can interpret. Figure 12-5 shows an example of a pseudocode algorithm for changing a lightbulb.

Programmers and customers can review a prototype and make changes before investing more time in the development of software.

Prototype the Solution For large projects, especially those that are not modifications to an existing project, the next step is to attempt to **prototype** the solution. This means creating a small semifunctional version of the solution to see if and how it works. In a Web site example, a programmer might first illustrate or develop just the visual

The Boss as Enabler?

YOU SEE THEM IN THE AIRPLANE, on the beach, at the restaurant: those suits looking ever so important, talking into their cell phone or logging onto their laptop. But those go-getters who seem to be wired 24/7 to stay on top of their profession may actually be Internet addicts—pushed into their illness by their employers.

Gayle Porter, an associate professor of Management at Rutgers University's School of Business, has completed a study that indicates a direct correlation between the extensive use of Blackberries and Internet addiction. Blackberries, sometimes dubbed "crackberries" because users become so quickly hooked, combine cell phone with wireless Internet. Many employees encourage—or even insist—that employees be accessible at all times. Porter suggests that such pressure to be technologically tied-in could be a source of stimulation that creates and feeds an addiction.

Employers often ignore signs of addiction—excessive use of wireless technology and stress-related illnesses—because the employee is getting the job done. But Porter concludes that businesses that currently provide programs to help workers deal with chemical or substance addictions may have to add IT addiction to the mix. And since courts have long recognized the duty of employers to provide a safe work environment for their employees, it is only a matter of time, Porter says, before employer pressure is cited in court as a factor in an employee's Internet addiction.

Source: Rutgers-Camden Research. "Employers, Beware: 'Techno Addicts' May Be More Liability than Boon," Rutgers, August 9, 2006. December 2006 <http://ur.rutgers.edu/medrel/viewArticle.html?ArticleID=5284>.

aspect of the main Web pages. Then, the customer requesting the Web site can review these pages. This provides the customer with something tangible to look at before deciding on the final design. It is much easier for the programmer to make changes at this point rather than later when the complex programming logic has already been coded. A prototype is a much less expensive way to make this determination than completing the entire project.

Implement and Test the Solution Once a successful prototype has been created, the next step is to fully solve the problem. This is where the program is actually coded. It is usually the most difficult and complex step in programming, especially if the problem is relatively new and unknown. Testing the solution frequently as each step of the algorithm is translated into a program helps ensure fast and accurate development.

Tech Demo 12-1
Extreme Programming

During the implementation phase, programmers sometimes learn that the planned solution simply is not working. This means that an error was made earlier in the problem-solving process and that it is time to return to the beginning and complete the steps again. It may be that another solution will have to be chosen from the original list of solutions. Documentation, such as a user's manual, is often written at this stage of the project.

The Russians Are Coming

OUTSOURCING SERVICE JOBS HAS BECOME a multibillion dollar business. More and more countries, including Vietnam, Botswana, and Argentina, are competing for a piece of the American pie, with India currently claiming the biggest slice. But the new face at the table competing for software development work from the United States is not a third-world country—it's our former nemesis Russia.

More than 250 global companies—including big players such as Boeing, Siemens, and the London stock exchange—have already hired Russian software developers. Estimates are that Russian workers will claim 5 percent of U.S. offshore programming revenue in the next few years.

Russian programming expertise lies in high-end, complex programming. The country has a highly educated workforce, with more than 50 percent of its students majoring in science, math, or computer science. UNESCO ranks it third in the world in terms of scientists and engineers per capita. Russia's homegrown IT sector, which receives strong government support, boasts almost 300 software companies.

The country's relative proximity to the United States gives the Russians a slight edge over India and Southeast Asia, as the end of the Russian programmer's workday overlaps the beginning of the East Coast's business day. Russian programmers command more money than their third-world counterparts, but still only one-fifth of U.S. pay rates.

Russian outsourcing promoters are also touting the country's current political stability and Westernized outlook on property laws, antipiracy regulations, and legal system as advantages over other rivals. The Cold War is over, and they want America to warm up to the idea of Russian programmers.

Sources: Loschinin, Dmitry. "Offshore Outsourcing: Why Russia?" *Rus-Soft*, March 20, 2006. December 2006 <http://www.russoft.org/docs/?doc=1123>; "Angling To Be the Next Bangalore," *BusinessWeek Online*, January 30, 2006. December 2006 <http://www.businessweek.com/magazine/content/06_05/b3969409.htm>.

The Evolution of Programming Approaches

While hardware technology improves at a very rapid rate, illustrated by Moore's Law, the doubling in speed and capacity of computers every 18 months, improvements in software development proceed much more slowly. The process is labor-intensive, requiring that highly trained individuals spend many expensive hours writing new programs and devising new methods. Despite this slower rate of progress, programmers have achieved some significant advances in software development during the past decades, including structured programming, modularity, object-oriented programming (OOP), and rapid application development (RAD).

Structured Programming

Programming was a new science in the early days of computing, and people wrote programs without applying rules to their development practices. Programmers were

People, rather than machines, power software development; improvements in software have come more slowly than improvements in hardware.

charting unknown territory, with no maps to guide them, so they tended to write whatever code they thought would work. **Structured programming** forced order onto this chaos, and presented guidelines for an organized, logical approach to programming that focuses on thinking at a higher level. Rather than simply executing lines one after another and writing code line by line, the programmer thinks in terms of structured groups of instructions. For example, if the program was designed to print out a bitmap image on the screen, such as a photo of a person's face on a Web page, the programmer builds a **routine** (sometimes called a function), which is a section of the program specifically to handle such images. The code for that routine is then broken down into steps, which in the case of a bitmap, would be to draw first one line of the grid, then the next, and so on. This process of creating groups of instructions as independent elements is the essence of structured programming.

Modules

One result of applying the divide-and-conquer approach to programming is the creation of **modular code**. Programmers work to create code modules that handle the separate components of a program. Like a single brick in a building, each module is a solid portion of a larger structure. This serves the purpose of making the code reusable, and also helps in tracking down the source of errors. When something goes wrong, programmers can usually assume that it is not the fault of the modular code because it has typically been tested and proved in use. Programs are described in terms of their **modularity**, which is a measurement of how well divided the source code is into individual modules. The higher the modularity of a program is said to be, the more reliable it will be.

Modular code also saves time because programmers don't have to "reinvent the wheel" every time they write new programs. An example of this efficiency would be a code module that reads data from a mouse and interprets user input

Digging in for Good

IF YOU WANT TO CATCH A WORM, you've got to think like a worm. David Aitel did, and as a result he has developed crime-fighting worms that spread quickly throughout a network to protect it from evil intruders.

Aitel is the chief technology officer at Immunity, Inc. His good worms are self-replicating computer programs that work their way into a computer by finding the same weak points that bad worms use. But instead of delivering destructive software, his worms close up the holes behind them.

The idea comes from the "patching worms" that hackers use to stop rival worms. But the hitch with all worms is that they are difficult to control once unleashed. After all, even a benevolent worm shouldn't enter a computer if uninvited. Aitel resolved the problem by programming his worms to only visit computers within a given range of IP addresses. His polite worms won't crawl just anywhere; they ask a central server for permission to enter.

Aitel calls his worms "nematodes" as in the parasites that gardeners let loose to kill pests in their flowerbeds. He has also developed a programming language called Nematode Intermediate Language (NIL) to help programmers develop their own do-gooder worms. Now, if they could only be programmed to eat up spam!

Source: Biever, Celeste. "Turning the Worm Secures the Computer," *NewScientistTech*, February 7, 2006. December 2006 <http://www.newscientisttech.com/channel/tech/mg18925376.500-turning-the-worm-secures-the-computer.html;jsessionid=GGIBIMMHCNED>.

Tech Demo 12-2
Creating Macros

by analyzing the mouse movements across the mouse pad. Once this module is finalized, it can be used as a component of any program requiring mouse input. A very simple type of modular code is a **macro** that is programmed into a spreadsheet or a document. A macro is a recording of steps to perform a repetitive activity. Although programming skills are not required to create macros, they are stored as a form of source code.

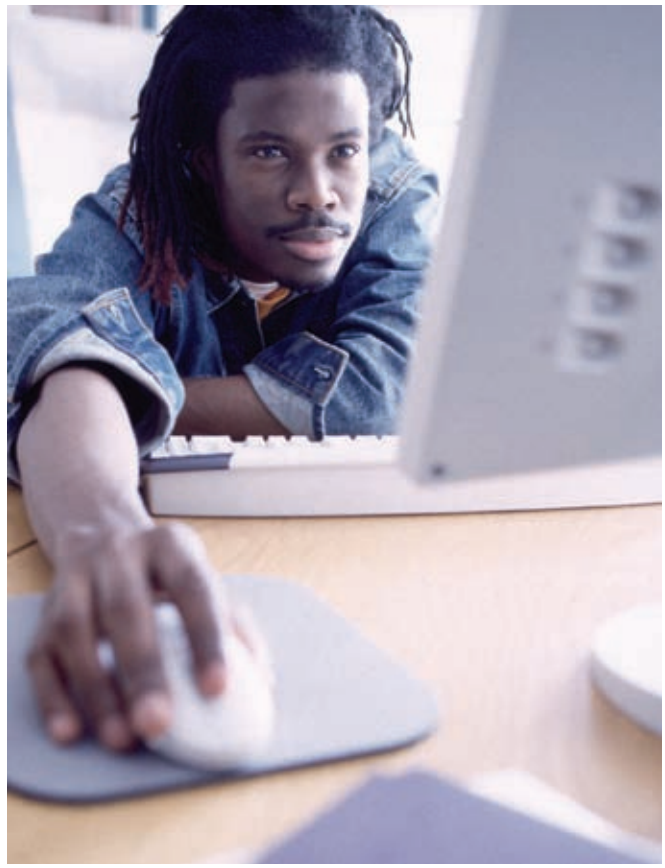
Object-Oriented Programming

Go to this title's Internet Resource Center and read the article titled "Object-Oriented Programming." www.emcp.net/CUT3e

Object-oriented programming (OOP) forces exacting rules on programmers. Essentially an extension of the modularity concept, OOP goes further by defining each module (called an **object**) with definite rules for interfacing and a protected set of variables. For example, if a program object were built to handle the time of day, it might hold data, including the seconds, minutes, and hours. Perhaps it also would have an a.m./p.m. flag and the functions that control the setting of the time would not allow illegal values. Thus if someone were to try to set the clock using an hour value of 27, the program would not allow it.

The protected variables are one of the key advancements in programming, as they allow a programmer to prevent data from being altered during program execution. A programmer working with a team sets up rules for the legal use of an object's data, thereby reducing the chances for error when someone unfamiliar with the code tries to use it. This technology is a great help with team development

projects because programmers can use the objects without needing to learn how they work. Simply understanding an object's interface enables programmers to use it.



Interpreting mouse movement as input is an example of modular code. Once this is programmed, other programmers can reuse it.

Rapid Application Development

Programmers began to experiment with **rapid application development (RAD)** techniques in the 1990s. Since labor costs constitute the major expense in software development, the primary focus of RAD was designing methods to reduce these costs by decreasing the time it takes to develop a project. Changes in the traditional approach to program development included eliminating labor-intensive written phases of the design stage by moving to early prototyping and by using higher-level languages. Visual Basic, Delphi, and other high-level languages with good interface capabilities are often used to aid in RAD. Programmers using RAD follow these guidelines:

- Use visual development (4GL) tools whenever possible.
- Rapidly prototype new projects in order to reduce redesign time.
- Approach coding with these priorities:
  - Use existing code first.
  - Buy someone else's existing code second.
  - Write new code last.

Another focus of RAD is to allow nonprogrammers to assist in a programming effort. Project managers, analytical chemists, and various types of engineers often need to use computers in specialized ways that require some programming. RAD techniques reduce the level of knowledge these individuals require.

Programming Development and Documentation Tools

Software developers have a variety of tools at their disposal to make their jobs easier. For example, at some point all programs must be translated from their original source code into machine code so they can be executed. Tools such as compilers, interpreters, and debuggers are used to handle this translation task and fix any errors that may result. Some of these tools are freeware, while others cost thousands of dollars. Their capabilities also vary greatly. When deciding which tools to use, a company or individual should consider a tool's cost, user-friendliness, and the performance of programs produced using the tool. Another important factor is whether or not the user is already familiar with the tool. Any training required must be included when calculating the tool's total cost. A sound purchasing decision can be made once all of these factors have been considered.

Compilers and Interpreters

Almost all computer languages come equipped with a compiler. A **compiler** is a program that translates programming language source code into machine code, the language computer chips use. A compiler reads an entire program prior to execution, and then displays a list of program errors that may be present. A compiler can only translate one language, and is specific to that language. A variety of compilers are available for most languages, each offering its own special features.

An **interpreter** differs from a compiler in that it translates instructions one-by-one as the source code is being executed, rather than all at once after reading through the entire program. Since an interpreter acts on just one line of instruction at a time, it identifies errors as they are encountered, including the line containing the error.

Compilers and interpreters have different strengths and weaknesses. Compiled programs tend to run much faster in their final form, and take up far less memory space than interpreted programs. Compilers also offer software manufacturers the advantage of allowing them to release only the unreadable machine code version of their product, while keeping the source code secret. This makes it popular with manufacturers of software distributed for retail sale.

Interpreted programs are easier to develop and test. Since there is no separate compilation step, programmers can click a button and execute the new program at any time. This ease of execution and testing speeds development, saving programming fees. Another positive feature is that interpreted languages can be distributed without a binary file. This is a popular solution with programmers, who can write special programs for the Internet that are many times smaller than comparable machine code files. The smaller size means these programs can be transmitted faster across the Internet. On the downside, interpreted programs run slower due to the time required to interpret lines one at a time.

Debuggers

A **bug** is a computer error. The term was originally coined by Grace Hopper, the inventor of COBOL. When a moth crawled into a computer's circuitry and caused it to malfunction, she called the error—quite literally—a "bug." A **debugger** is a software tool that helps programmers find errors quickly. Debuggers also allow programmers to examine closely what is happening when a program runs. In most cases debuggers are an integral component of compilers and interpreters, just as a spellchecker typically is integrated with a word processor.

Documentation Tools

Proper documentation is one of the key elements of software development. **Documentation** consists of written notes that explain how a program works. Several tools exist to help with the creation of documentation, including flowcharts, CASE tools, and comments.

Flowcharts A **flowchart** provides a visual diagram of an algorithm. Instead of using pseudocode, a programmer can draw a flowchart showing the logic of a program with symbols that represent types of action (see Figure 12-6). Today's professional programmers rarely use flowcharts, as most software programs are too

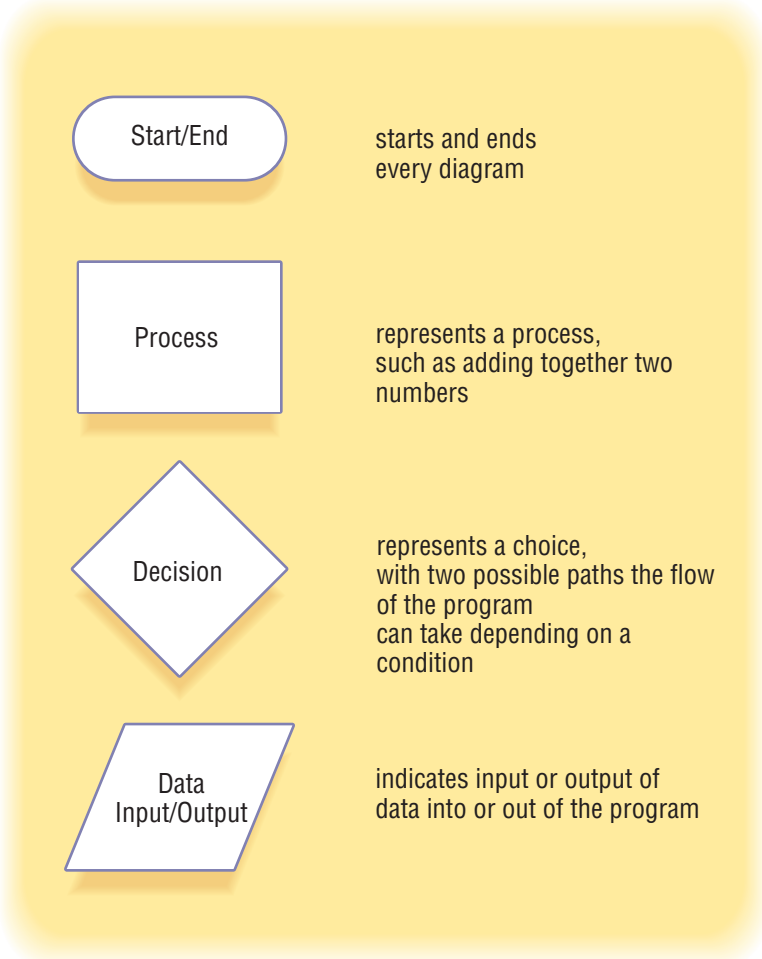

Start/End
starts and ends every diagram
Process
represents a process, such as adding together two numbers
Decision
represents a choice, with two possible paths the flow of the program can take depending on a condition
Data Input/Output
indicates input or output of data into or out of the program

Figure 12-6 Flowchart Symbols
These symbols are used in flowcharts to represent the logic of a program.

complex to be represented graphically in every detail. However, beginning programmers still use flowcharts to help visualize their algorithms and understand their programs before they attempt to write code. Figure 12-7 displays an example of a flowchart using standard flowchart symbols.

Go to this title's Internet Resource Center and read the article titled "Working with Pseudocode and Flowchart Documents." www.emcp.net/CUT3e

CASE Tools CASE tools, as described in Chapter 10, help a programming team schedule and coordinate its operations. Similar to a suite of applications, some typical CASE tools include a scheduler for building Gantt charts and software that assists with creating documentation, including the data dictionary.

Comments A **comment** is an informational message inserted into program source code. Comments are normally used to explain source code to later readers, like notes scribbled in the margins of a book. Comments can be interspersed throughout a program to help others understand how the program works, and to provide the reasoning behind programming decisions. They are written in English rather than in a programming language (see Figure 12-8). Comments have a special designation, which varies by language, so the compiler/interpreter knows to ignore them.

Programming Errors

Programmers actually spend a large portion of their time fixing errors, or bugs. There are several types of errors that programmers must isolate and fix one at a time to get a program working. The main error types are:

- syntax errors
- logic errors
- run-time errors
- style errors

Errors often occur because a programmer was sloppy during the programming process, instead of paying careful attention. In programmer circles, this is referred to as **hacking code**. Hacking in this sense does not refer to breaking into security systems. If a programmer is hacking code, he or she is writing code without carefully planning and structuring the programs. Code written this way is much more likely to have errors in it. Among programmers, calling a colleague a hacker is a mildly derogatory term referring to lazy habits and sloppy work.

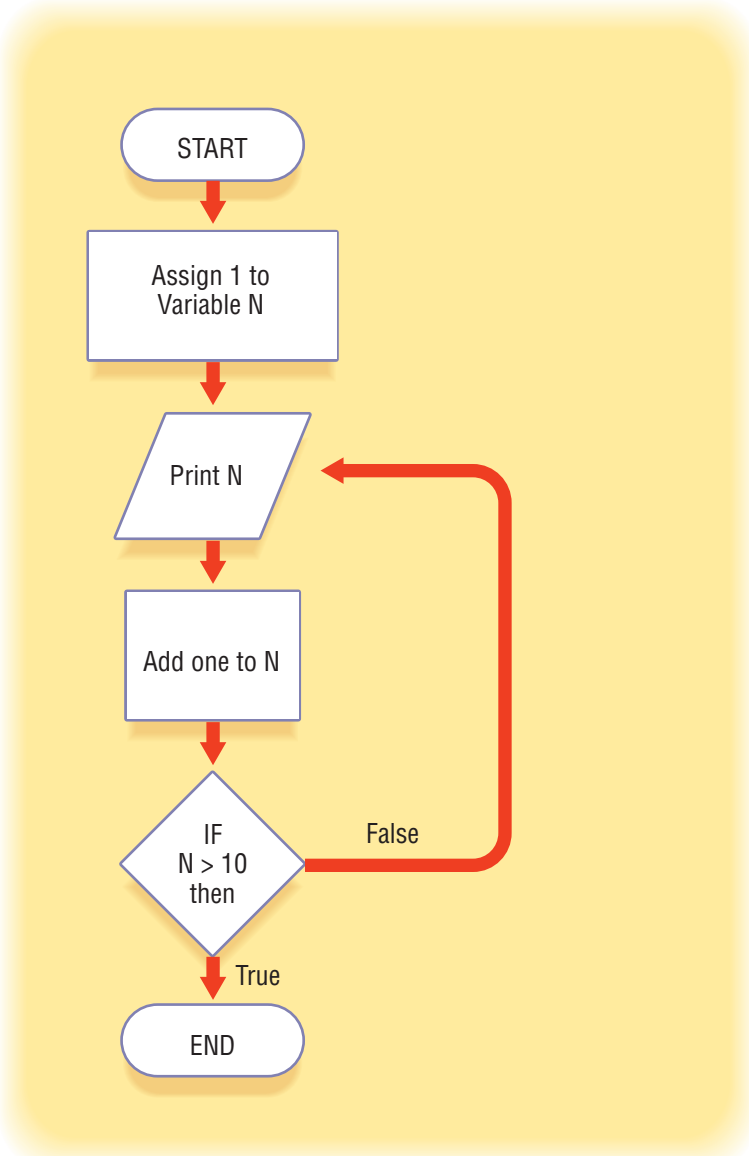


Figure 12-7 Algorithm Flowchart
Flowcharts help programmers visualize the steps in a software program.

Syntax Errors

As explained earlier in this chapter, syntax errors, like grammar errors, are usually due to typing mistakes or a misunderstanding of the rules of a language. Computer languages are very structured and require that programmers use them flawlessly to create a working program. To use an analogy, if a program were created to read a paragraph in a newspaper, it might require that every sentence start with a capital letter and end with a period so that it could recognize them. Any deviation from these requirements would prevent the computer from recognizing a string of words as a sentence, causing it to reject the newspaper and demand that the error be fixed.

Logic Errors

A **logic error** occurs when a program's syntax is correct, but the program instructs the computer to perform an action incorrectly. For example, if a program was meant to add a column of numbers and instead multiplied them because of an incorrect instruction, this would be considered a logic error. The program might run, but it won't do

```
int x;      // creates an integer variable called x
x = 12;     // sets the value of the variable x to 12
cout        << x << endl; // prints the value of x and goes
                          // to the next line
```

Figure 12-8 C++ Source Code Fragment Containing Comments
Comments help other programmers understand how the program works. A set of forward slashes (//) indicates the start of a comment and a hard return indicates the end of the comment.

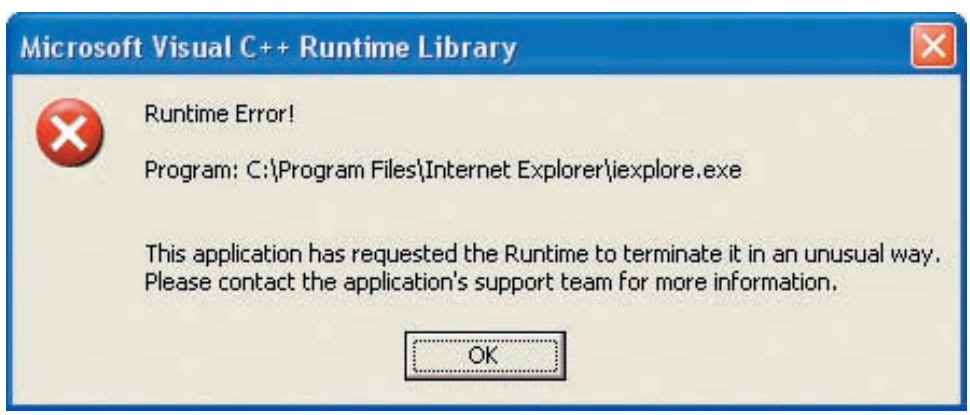
what the programmer intended it to do. This type of error is very common and is the most difficult type of error professional programmers must deal with.

Run-Time Errors

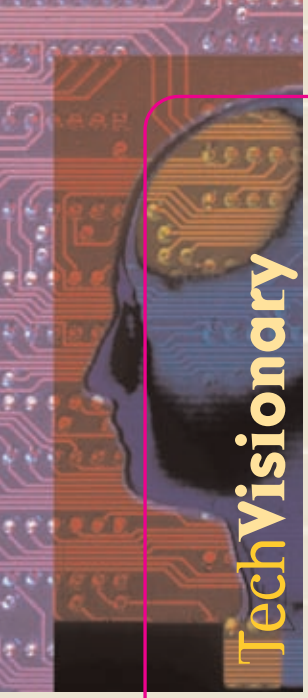
A **run-time error** refers to mistakes that occur when an application is running. A **crash bug** is one of the most dreaded types of run-time errors because it causes a program to stop running, or crash. If a program crashes it is useless, so programmers usually attempt to solve this kind of bug first. Another common cause of run-time errors is an **infinite loop**, which is a programming mistake that causes a program to perform the same set of instructions over and over again without any way of stopping. Infinite loops cause computers to "lock-up" or "freeze," a state in which they no longer respond to input from the mouse or keyboard.

Style Errors

A **style error** in programming code is similar to bad grammar in a term paper. A term paper may still be readable despite bad grammar, but the bad grammar will


Microsoft Visual C++ Runtime Library
Runtime Error!
Program: C:\Program Files\Internet Explorer\iexplore.exe
This application has requested the Runtime to terminate it in an unusual way.
Please contact the application's support team for more information.
OK

A crash bug, a type of run-time error, is one of the most dreaded programming errors because it causes the program to crash.



TWO FAMOUS NAMES IN THE PROGRAMMING WORLD

Wirth and Dijsktra

DR. NIKLAUS WIRTH

In the 1960s, several computer scientists worked on extending Algol, an algorithmic programming language. One of these was Dr. Niklaus Wirth of the Swiss Federal Institute of Technology (ETH-Zurich), who was a member of the original group that created Algol. Later, in 1971, he published his specification for a similar language that he named Pascal, after the seventeenth-century French philosopher and mathematician. Pascal soon became the standard teaching language in universities across the globe, and remained the language of choice for students of computer science until the 1990s.

Pascal is a highly structured, high-level language that is fairly simple to learn. It was popular for teaching purposes primarily because it forced programmers to program in an organized fashion and allowed advanced data structure definition. As a learning tool, it taught good programming habits to programmers who would later move on to more freewheeling professional languages, such as C. Pascal greatly influenced the design of later languages such as Visual Basic and Ada.

Dr. Wirth was born in Switzerland in 1934. He obtained his PhD from the University of California–Berkeley in 1963 and taught computer science at Stanford University. Today, he teaches at the Federal Institute of Technology in Zurich, Switzerland.

DR. EDSGER DIJSKTRA

Dr. Edsger Dijsktra, a native of the Netherlands, contributed much in the way of new algorithms to computer science. Many advanced textbooks in the field contain references to his works, which he published prolifically in the 1960s and 1970s. He is also famous as a champion of the organized development of structured code. His theories on error reduction largely focused upon not putting the errors into the code in the first place. "If debugging is the process of removing bugs," he said, "then programming must be the process of putting them in."

Dijsktra is possibly most famous for a 1968 article entitled "GO TO Statements Considered Harmful," in which he introduced the concept of structured programming and developed standards for constructing computer programs in an organized fashion. His work usually was accompanied with a sharp wit, as is evident in another of his infamous statements: "Teaching COBOL ought to be regarded as a criminal act."

Source: Association for Computing Machinery. December 2006 <http://www.acm.org>.

detract from a reader's enjoyment, and most likely the student's grade. In the same way, a program may still run with style errors, but programmers may be unhappy with the way the program was written.

One style error programmers commonly make is called **dead code**. This is code that is commented out, which means that it is marked with comments to notify the compiler to skip and ignore it. Dead code makes it hard to read source code, as it appears to be part of the program but actually has no effect on the way a program runs.

The Software Development Life Cycle

The **software development life cycle (SDLC)** is the term used to describe the phases involved in the process of creating, testing, and releasing new software products. This cycle is similar to the process used in developing information systems, except that in this case the cycle focuses on the creation and release of a software program, not the development of a customized information system. The SDLC is repeated every time a new version of a program is needed. As shown in Figure 12-9, the phases in the software development life cycle include:

- proposal and planning
- design
- implementation
- testing
- public release

Proposal and Planning

In the proposal and planning phase of a new software product, software developers will describe the proposed software program and what it is supposed to accomplish. In the case of existing software, the proposal and planning stage can be used to

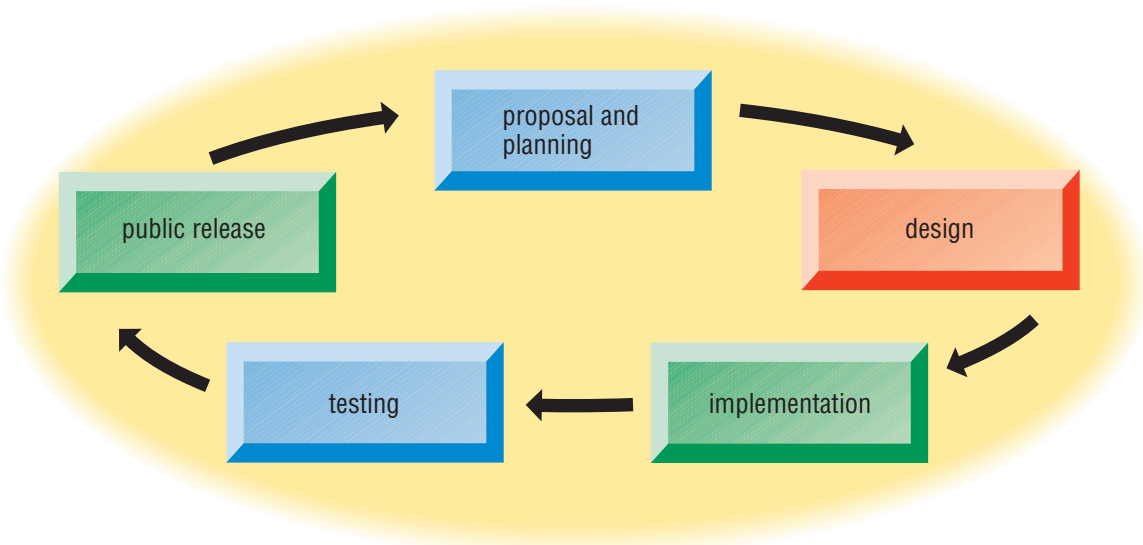


Figure 12-9 The Software Development Life Cycle
The software development life cycle involves planning, designing, implementing, testing, and release of application software.

describe any new features and improvements. Older software programs are often revised to take advantage of new hardware or software developments and to add new functions or features.

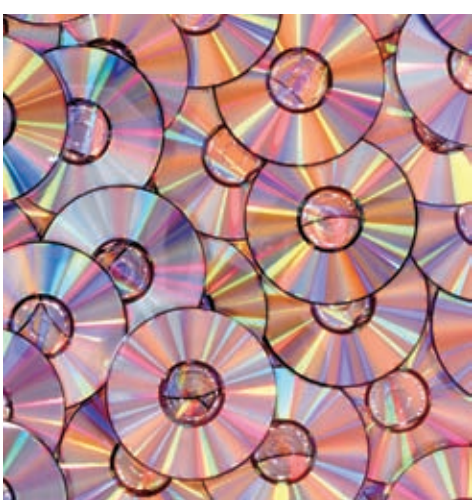
Design

Developers are ready to begin the design process once the decision has been made to create or upgrade a software program. This step produces **specifications** documenting the details of the software to be written by programmers. Developers use the problem-solving steps described earlier in this chapter to determine appropriate specifications.

Implementation

The implementation phase of the software life cycle is usually the most difficult. Development teams often spend late nights and weekends writing code and making it work. If the planning and design effort has been successful, this phase should go well, but unanticipated problems inevitably crop up and have to be solved. The end result of the implementation phase is the production of a prototype called an **alpha product**, which the development team uses for testing purposes. The alpha product can be revised to incorporate any improvements team members suggest.

Testing



Software programs are tested carefully before the master is released for duplication.

A quality assurance (QA) team usually develops a **testing harness**, which is a scripted set of tests that a program must undergo before being considered ready for public release. These tests might cover events such as very large input loads, maximum numbers of users, running on several different supported platforms, and simulated power outages.

Once testing is finished, a **beta version** of the software program is created for testing outside of the development group, often by a select group of knowledgeable consumers. Any suggestions they make can be used to improve the product before it is released to the general public. Once the beta version is finalized, the user manual can be written or updated. At this point the software developer would send the master CDs to duplicators for mass production.

Public Release

When the product is deemed ready for widespread use, it is declared "gold" and released to the public. A **gold release** is also referred to as a "generally available" release of the software. The software life cycle now goes back to the beginning phases as software developers think of new ways to improve the product. The cycle continues until a new version (revision) of the original product is ready for release. Software usually has major releases and minor revisions. Versions of software typically represent the major release and minor revision. For example, Adobe Acrobat Reader 6.01 is a newer minor revision of Acrobat Reader 6.0, which is a major release after Acrobat Reader 5.1.

Major Programming Languages

The very first programming language was machine code, since it is the language that computers use for their internal operations. Programming in machine code is difficult because it consists entirely of numbers. Assembly language was invented to make programming easier, by attaching symbols and words to represent the numbers in machine code. Since the appearance of the first assembly language, hundreds of high-level programming languages have been created, but only a few are used extensively for professional work today. Some of the most popular programming languages currently in use include C, C++, Visual Basic, Java, and script languages such as the many variants of HTML. Some languages dating back to the 1960s, including COBOL and FORTRAN, are still used by large organizations such as the IRS, the Social Security Administration, the federal banking system, the military, and universities. Table 12-1 compares characteristics of some of the most popular programming languages.

Machine Code

As previously defined, machine code is the computer programming language that computers actually read and interpret. The code is written using a series of binary strings. A **binary string** is a sequence of binary symbols, such as 1s and 0s. Computers can read and act on this code since computers "think" using the binary system. Programming is rarely done using machine code because it is difficult for anyone to memorize the long binary strings that comprise machine code.

Assembly Language

Assembly language uses symbols and words to represent the elements of machine code, making it possible for programmers to memorize them. Computers cannot read assembly language, so it must be converted to machine code by a compiler before it can be used.

Compared with other computer languages, assembly languages run the fastest and use the least memory. The chief drawback to using them is that assembly language programs are difficult to write, and therefore development times are lengthened. Assembly languages are used wherever speed and memory are at a premium, and maximum possible efficiency is desired. Embedded computers such as cell phones, calculators, and other electronic devices are often programmed using assembly language.

Table 12-1 Commonly Used Programming Languages

Language	Compiled	Interpreted	Object-Oriented
C	Yes	No	No
C++	Yes	No	Yes
Java	Yes	Yes	Yes
Visual Basic	No	Yes	No

Binary and Hexadecimal Coding

TO A COMPUTER, all is 1s and 0s, the binary numbers of machine language that a computer understands. A human being, however, has a difficult time writing a program or reading output in machine language. Therefore, programmers have looked for ways to simplify binary coding. One method is to use hexadecimal, or base 16, numbers. The decimal system makes use of ten characters, 0–9. The hexadecimal system adds another six characters, the capital letters A–F, to represent the numbers 10–15.

Decimal
0 1 2 3 4 5 6 7 8 9
10 11 12 13 14 15

Hexadecimal
0 1 2 3 4 5 6 7 8 9
A B C D E F

A single hexadecimal digit can represent any binary number that contains four digits or less, from 0 to 1111. For example, the ASCII code for the symbol ½ is the byte 10101011. This number can be broken into two four-digit numbers, or nibbles, 1010 (10 in decimal) and 1011 (11 in decimal), that can then be represented by two hexadecimal numbers: AB. Thus the ASCII code for ½, in hexadecimal, is AB, a number far easier to remember than 10101011.

Programmers have also made their job easier by creating programming languages that are less cumbersome to read, write, and understand than machine language. Assembly language is a language that is still not far removed from machine language. Basically, assembly language consists of symbols for the binary commands that make up the instruction set, or list of executable commands, in the computer's processor. Instructions written in assembly language are read into a program called an assembler, that translates the instructions into machine language.

Source: Shepherd, Robert D. *Introduction to Computers and Technology*, Paradigm Publishing: October 1997.

COBOL

An acronym for COmmon Business-Oriented Language, **COBOL** is used chiefly for business applications by large institutions and companies. It was developed in 1960 by Dr. Grace Hopper. Although its use is diminishing, many schools still require students to study COBOL. Originally designed to be an English-like language for handling database processing, COBOL has changed little since its creation. It is a slow and cumbersome language, but it has a large body of existing code and many programmers are familiar with it.

RPG

An acronym for Report Program Generator, **RPG** is commonly used in business environments, particularly in programs for IBM's AS/400 mainframe. It is slow

and inefficient, but simplifies the coding of database applications, and it has the advantage of being familiar to many programmers. Many midrange and mainframe computers rely heavily on RPG.

FORTRAN

FORTRAN stands for "FORmula TRANslator" and for many years was the language of choice for math, science, and engineering projects. Created in 1957, FORTRAN dominated scientific and engineering programming for decades. FORTRAN is still in use today in factories and laboratories, although it is not as common as COBOL.

BASIC

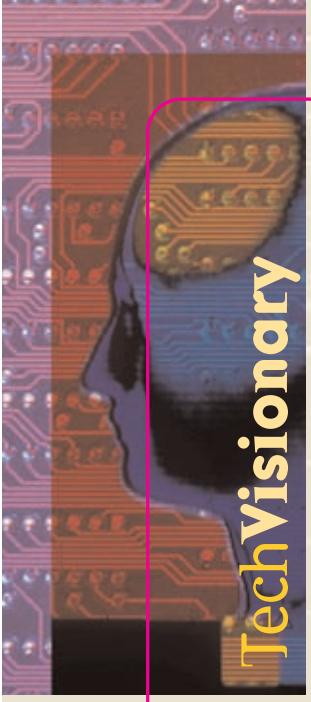
"Beginner's All-purpose Symbolic Instruction Code," or **BASIC**, is a high-level language that is friendlier and more natural than earlier languages such as COBOL and FORTRAN. Originally designed as a teaching tool in the 1960s, BASIC is still used professionally in an updated form, often Visual Basic. BASIC is an interpreted language, meaning that it runs slowly. This drawback is somewhat compensated for by the fact that development in BASIC tends to go more quickly for the programmer. The original BASIC only supports command-line interfaces, rather than graphic point-and-click technology. A comparison of the BASIC sample code with C++ code (see Figure 12-10 and Figure 12-11) shows why BASIC is considered easier to understand.

```
10  Rem This code sample prints out 1 through 10 on the screen.
20  For Count = 1 to 10
30  Print Count
40  Next Count
50  End
```

Figure 12-10 Example of BASIC Source Code
This sequence of code will print the numbers 1 to 10.

```
#include <iostream.h>
void
main()
{
      int count; // Create a variable called "count"
      for (count = 1; count <= 10; count++)   // Loop 10 times
            cout << count << endl;       // Print 10 values
}
```

Figure 12-11 Example of C++ Source Code
This sequence of code will print the numbers 1 to 10.



INVENTOR OF COBOL

Grace Hopper

THE NAME OF DR. GRACE HOPPER could be associated with several topics in the field of information technology, considering the depth and breadth of her contributions. Born in New York City in 1906, she led an interesting life that spanned careers in the military, academia, and business.

Hopper graduated Phi Beta Kappa from Vassar College in 1928 with a degree in mathematics and physics. Two years later, she received her master's degree in mathematics from Yale University. She earned a PhD from Yale in 1934 and taught at Vassar College from 1934 until 1943, when she joined the Navy. After finishing her training at Midshipman's School for Women, Hopper was assigned to the Bureau of Ordinance Computation at Harvard University, where she became the third programmer of the Mark I, the world's first general-purpose, automatic digital computer. The Navy used the Mark I to calculate aiming angles for its guns, and Hopper and her assistants often were on call 24 hours a day, inputting information and managing the system. It was while working on the Mark II that Hopper became credited with coining the term "bug." According to computer lore, she and her team discovered a moth trapped in the Mark II's circuitry and surmised that it was the cause of the system shutting down.

Hopper left the Navy at the end of World War II, but remained at Harvard as a civilian research fellow in engineering sciences and applied physics. In 1949, she joined the Eckert-Mauchley Computer Corporation (which became the Sperry Corporation) that later developed the first commercial computers, UNIVAC I and II. She also led a team that designed the first compiler, to be used for typical business tasks such as payroll calculation and automated billing.

Hopper soon realized a need for a standardized, universal computer language. In 1959, she and her staff developed the first specifications for the programming language COBOL. The language uses English-like phrases to describe programs that handle business-type operations. For several years, COBOL was the language of choice for database development in many corporate computer departments and that preference continues to be the case. To this day, the administrative functions of the U.S. government can be said to run on millions of lines of COBOL.

In 1966, Hopper returned to active duty with the Navy to help its IT staff standardize their high-level computer languages. She also helped develop programs to convert nonstandard COBOL languages into the standardized version. In 1986 at age 80, Rear Admiral Grace Hopper retired on the deck of the *USS Constitution*. She rejoined civilian life by becoming a senior consultant to Digital Equipment Corporation.

Hopper received numerous honors over the course of her lifetime, including many honorary doctorates, prestigious awards from computer organizations, and having a Navy ship named after her. However, she maintained that her most valuable and satisfying achievement was teaching. Grace Hopper died in 1992.

Sources: "Grace Hopper," *San Diego Supercomputer Center*. December 2006 <http://www.sdsc.edu/~woodka/hopper.html>; "Grace Murray Hopper," *Agnes Scott College*. December 2006 <http://www.agnesscott.edu/lriddle/women/hopper.htm>.

Visual Basic

Tech Demo 12-3
Visual
Programming

Visual Basic (VB), developed by Microsoft Corporation in the early 1990s, is currently the professional's language of choice for developing software prototypes, or for developing custom interfaces for Windows platforms. Programs written in VB are not generally mass-marketed. They are quick and easy to develop, but run slowly and are demanding of resources such as RAM and disk space. Visual Basic supports graphic interfaces, unlike the original BASIC of the 1960s. Because of this feature it is easier to use compared with other languages such as C or C++.

C

C was originally developed at Bell Labs in the early 1970s by Dennis Ritchie for the early UNIX operating system. The language was named C because it builds on an earlier language called Basic Combined Programming Language (BCPL), or "B." It is a compromise between high-level and low-level languages, containing components of both BASIC and assembly language, but is still considered a high-level language. C programs aren't as easy to read as BASIC, but they produce programs that run considerably faster and use less space. C isn't quite as fast as assembly code when executing, but it is much faster than BASIC.

C++

C++ is an adaptation of C. More specifically, it is a superset of C, and any C program should run without problems as a C++ program. This means that a compiler that compiles and understands a program written in C++ will also understand a program written in C, as C++ is the same as C, but with added features such as object-oriented programming. Most professional software sold today is written in some form of C or C++. Figure 12-11 shows a section of C++ source code that prints out the numbers 1 through 10.

C#

Pronounced "See-Sharp," **C#** is a modern object-oriented language derived from C++ and Java. It combines the productivity of the simplistic Visual Basic with the power of the C++ language. C# programming allows the use of features in the Microsoft .NET framework, C, and Microsoft's Component Object Model (COM).

Java

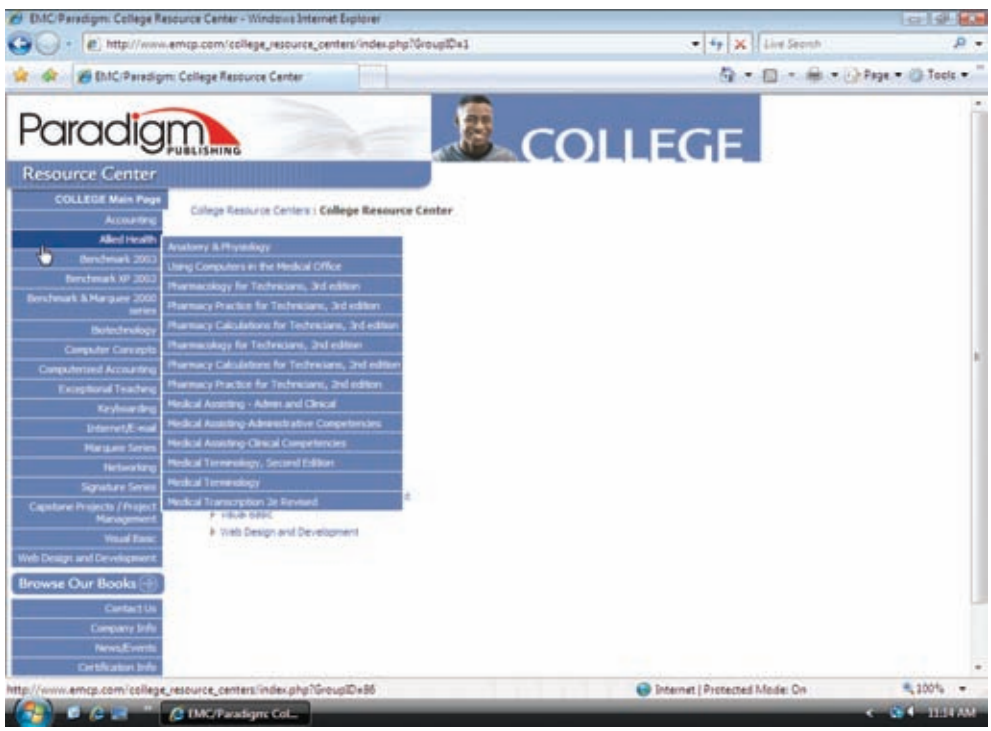
Sun Microsystems's **Java** programming language has become widely used on the Internet as a script language for specialized Web page applications. Composed of small applications (each one called an **applet**) that can run on all types of computer operating systems, the program was created to facilitate communication on the Internet among the wide variety of user platforms. Central to the software is the **Java Virtual Machine (JVM)**, which converts the general Java instructions into commands that a device or computer can understand. A JVM is currently built into most Web browsers. It works with the operating system on embedded chip devices such as smart cards or Web-enabled cellular phones, or it may be integrated directly into them. Java is very similar to C++, but is designed to run either as an interpreted

script transmitted over the Internet, or as a compiled (machine code) executable program. This flexibility and its **cross-platform compatibility** to run on nearly any operating system have made it very popular with programmers around the world.

Scripting Languages

A **scripting language** is an interpreted language that is relatively easy to learn and use. A scripting language is a **nonprocedural language**, meaning that it explains what the computer should do in English-like terms, but not precisely how the computer should do it. Web pages are usually built with script languages such as Hyptertext Markup Language (HTML) and JavaScript. Advanced wizards and editors in programs using script languages allow people to build Web pages without knowing the underlying HTML code. Users of these systems may not even realize they are programming. There are a variety of improved Web page capabilities provided by newer versions of HTML, including DHTML, XHTML, XML, and WML. Other popular script languages used in building Web pages are **perl** (practical extraction and report language), **VBScript** (a scripting language loosely based on Visual Basic), and **JavaScript** (developed by Sun Microsystems). See Figure 12-12 for a sample of JavaScript. JavaScript can run as an interpreted script over the Internet or as a compiled executable program.

DHTML **Dynamic HTML (DHTML)** allows Web page designers to alter the contents of a Web page screen dynamically. Dynamically refers to the fact that viewer actions can be used to change the contents of a Web page. DHTML also displays faster than older pages built with HTML. Pages created using DHTML are usually higher quality in appearance and are more graphically responsive to user actions. For example, buttons may light up when the mouse passes over them, or graphics might change depending on the position of the mouse. DHTML uses an object-oriented programming model called **document object model (DOM)**.



Web pages using DHTML can change graphically when the mouse pointer moves over the hotspot on the screen.

```
<HEAD>

<SCRIPT LANGUAGE="JavaScript">

<!--Comment: This JavaScript Code will cause three buttons to appear that
can change the background color when clicked.-->

<!--Begin
function newcolor(aColor)
{
document.bgColor=aColor;
}
// End-->
</script>

</HEAD>


<BODY>

<div align="left">
<form>
 <input type="button" value="Red" onclick="newcolor('red');">
 <input type="button" value="White" onclick="newcolor('white');">
 <input type="button" value="Blue" onclick="newcolor('blue');">
</form>
</div>

</BODY>
```

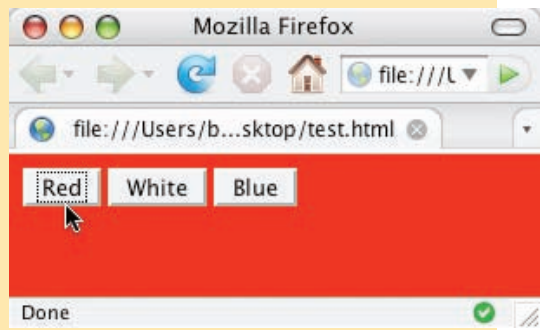


Figure 12-12 Example of JavaScript
This sequence of code directs a browser to display buttons that users can click to change the background color within the Web browser window.

XHTML **Extensible HTML (XHTML)** is another improvement in Web page programming, achieved through the upgrading of the underlying script language. XHTML lets Web designers use HTML and **Extensible Markup Language (XML)** together. XML allows Web pages to separate their content from the format supported by a Web page. Thus screens with different sizes and resolutions can display the same Web page content without the Web designer having to write specialized versions for each. For example, a notebook or handheld computer can show the same Web information in different formats that fit the screen of each device.

Wireless Markup Language **Wireless Markup Language (WML)** was created specifically for small systems such as cell phones and handheld computers. The language allows designers to build pages that work for small screens with limited display and input capabilities.

OnTheHorizon

AS WITH NEARLY EVERY AREA OF INFORMATION TECHNOLOGY, the Internet affords capabilities and opportunities that are influencing the path of programming and application design. Among the new directions making programming easier are the use of artificial intelligence to develop software, and programming tools that will let non-programmers create their own applications.

Subtext: Making Life Easier for Programmers

Programming is mentally challenging work. Donald A. Norman, Professor of Electrical Engineering & Computer Science, Psychology, and Cognitive Science at Northwestern University, has identified two issues that make programming difficult, and termed them the "Gulfs of Evaluation and Execution." The "Gulf of Execution" refers to the difficulty in transforming desired goals into programming code, and the "Gulf of Evaluation" refers to the difficulty in evaluating in real-time whether or not coding goals are being met. An experimental programming language known as Subtext hopes to overcome these obstacles and make programming much easier by incorporating cut and paste features with the live text found in spreadsheet programs. The cut and paste aspects of Subtext aim to facilitate the realization of programming goals, while live text will make evaluation easier by enabling instant feedback. If the Subtext project meets it goals it could serve as a new programming model that will simplify work for future programmers.

Software Writing Software

The development of robotics systems for building smaller computer components is the driving force behind the miniaturization of computers. Human beings could never fashion these chips and boards by hand, no matter how skilled they are. Robotics systems improve each year, which in turn allows them to build even better machines. This means that the computers are continually improving themselves. This ongoing cycle causes computers to grow in capacity and speed, while shrinking in size and cost every year.

Unfortunately, a similar effect has yet to be seen in the software development area. Software is still written manually, and the human mind powers the entire process. Since human minds aren't doubling their capabilities every two years, the rate of improvement is slow. However, if science gets to the point of creating artificially intelligent machines that are capable of writing their own software, we may yet see the dramatic improvements in software that we are now seeing in hardware. This is one of the major topics discussed in the next chapter, *Multimedia and Artificial Intelligence*.

Instant Programming

The latest trend in programming tools are platforms that allow non-programmers to drag-and-drop ready-made modules into complete programs meeting their application needs. The ability to create tailored business applications without the need for expensive and time-consuming IT team participation is sure to attract the attention of the business community. Lower prices for hardware and bandwidth, the increasing compatibility of open-source systems, and new technologies are making these tools increasingly technically and financially viable. One analyst predicts that it will take about three more years for these platforms to fully realize their potential, and says that using them will be as easy as snapping together Lego bricks.

Chapter Summary

For an interactive version of this summary, go to this text's Internet Resource Center at www.emcp.net/CUT3e. A Spanish version is also available.

What makes up a programming language?

A **program** is a set of instructions telling a computer how to perform various tasks, and programming is the act of creating these instructions. Large programming projects are much like engineering efforts. They are too big to be completed by a single individual, so they require a programming team. Software engineers typically coordinate programming projects working with programming teams. Programmers create programs using **programming languages**. Programming languages contain smaller vocabularies than human (natural) languages, and are much less complex. Programming language **syntax** (the rules for stringing together language elements) also tends to be less complex than human language syntax. Computer languages can be classified as being **low-level language** or **high-level language**. They can also be classified by language generation. Programming language text files are called **source code**, which requires special training to learn. **Coding** is a term programmers use to refer to the act of writing source code. The four main computer language programming elements are **variables**, **executable statements**, **looping**, and **decision statements**.

What problem-solving techniques are used in programming?

Programming is a problem-solving process. Problem-solving is as much an art as a science, but there are techniques for solving problems using logical processes, such as the **divide-and-conquer approach** and **problem-solving process**.

How has the programming approach evolved?

Significant milestones in the history of software development include **structured programming**, **modularity**, **object-oriented programming (OOP)**, and **rapid application development (RAD)**.

What development and documentation tools do programmers use and why are these tools important?

A variety of tools are available to software developers to make their jobs easier. A **compiler** translates programming language source code into machine code prior to execution, and then displays a list of program errors that may be present. An **interpreter** differs from a compiler in that it translates instructions one by one as the source code is being executed. A **debugger** is a software tool that helps programmers find errors quickly. **Documentation** consists of written notes that explain how a program works. Several tools exist to help with the creation of documentation, including **flowcharts,** CASE tools, and **comments**.

What types of programming errors are found in programs?

The main types of errors programmers must isolate and fix, one by one, to get a program working are **syntax errors, logic errors, run-time errors,** and **style errors**. **Hacking code** means that programmers are writing code without carefully planning and structuring their programs.

What are the steps in the software development life cycle?

The **software development life cycle (SDLC)** is used in the creation, testing, and release of new software products. The cycle includes the proposal and planning, design, implementation, testing, and public release phases.

What are examples of major programming languages?

The very first programming language was **machine code**. **Assembly language** was invented to make programming easier by attaching symbols and words to represent the numbers in machine code. Since the appearance of the first assembly language, hundreds of high-level programming languages have emerged.

Some programming languages date back to the 1960s, including **COBOL** and **FORTRAN**, but are still used by large organizations such as the IRS, the Social Security Administration, the federal banking system, the military, and universities. Some of the most popular programming languages currently in use include **C, C++, C#, Visual Basic, Java,** and **scripting languages**.

KeyTerms

Numbers indicate the pages where terms are first cited in the chapter. An alphabetized list of key terms with definitions (in English and Spanish) can be found on the Encore CD that accompanies this book. In addition, these terms and definitions are included in the end-of-book glossary.

Programming Concepts

program, 541
programming language, 541
source code, 541
coding, 541
syntax, 541
syntax error, 541
low-level language (machine code), 542
high-level language, 542
programming language generation, 542
variable, 543
executable statement, 543
looping, 543
decision statement, 545
if-then statement, 545

Problem-Solving Techniques

problem-solving process, 546
divide-and-conquer approach, 546
top-down design, 547
algorithm, 550
pseudocode, 550
prototype, 550

The Evolution of Programming Approaches

structured programming, 553
routine (function), 553
modular code, 553
modularity, 553
macro, 554
object-oriented programming (OOP), 554
object, 554
rapid application development (RAD), 555

Programming Development and Documentation Tools

compiler, 556
interpreter, 556
bug, 556
debugger, 556
documentation, 556
flowchart, 556
comment, 557

Programming Errors

hacking code, 558
logic error, 558
run-time error, 559
crash bug, 559
infinite loop, 559
style error, 559
dead code, 561

The Software Development Life Cycle

software development life cycle (SDLC), 561
specifications, 562
alpha product, 562
testing harness, 562
beta version, 562
gold release, 562

Major Programming Languages

binary string, 563
assembly language, 563
COBOL, 564
RPG, 564
FORTRAN, 565
BASIC, 565
Visual Basic (VB), 567
C, 567

C++, 567
C#, 567
Java, 567
applet, 567
Java Virtual Machine (JVM), 567
cross-platform compatibility, 568
scripting language, 568
nonprocedural language, 568
perl, 568
VBScript, 568
JavaScript, 568
Dynamic HTML (DHTML), 568
document object model (DOM), 568
Extensible HTML (XHTML), 569
Extensible Markup Language (XML), 569
Wireless Markup Language (WML), 569

Chapter Exercises

The following chapter exercises, along with new activities and information, are also offered in the Internet Resource Center for this title at www.emcp.net/CUT3e.

Tutorial > Exploring Windows

In Tutorial 12, you will learn some tips for playing music or movies on your computer.

Expanding Your Knowledge > Articles and Activities

Visit the Internet Resource Center for this title at www.emcp.net/CUT3e, read the articles related to this chapter, and complete the corresponding activities. The article titles include:

- Tutorial 12-1: Logic Tables and Control Structure
- Tutorial 12-2: Linear Programming and Simplex Method
- Tutorial 12-3: Object-oriented Programming
- Tutorial 12-4: Working with Pseudocode and Flowchart Documents

Terms Check > Matching

For additional practice, go to the Internet Resource Center for this title at www.emcp.net/CUT3e for a chapter crossword puzzle.

Write the letter of the correct answer on the line before each numbered item.

a. Visual Basic
b. compiler
c. if-then statements
d. looping
e. machine code
f. RAD
g. pseudocode
h. syntax
i. crash bug
j. interpreter

____ 1. A severe type of run-time error.
____ 2. Translates source code into machine code.
____ 3. Used by software programs to repeat statements.
____ 4. The lowest level of programming language.

____ 5. A listing of English-like instructions that forms an outline describing a program, but it does not compile or run the program.

____ 6. The rules for stringing together elements in a computer language.

____ 7. A program that reads source code and executes the statements immediately, without first creating a binary interpretation of the source code.

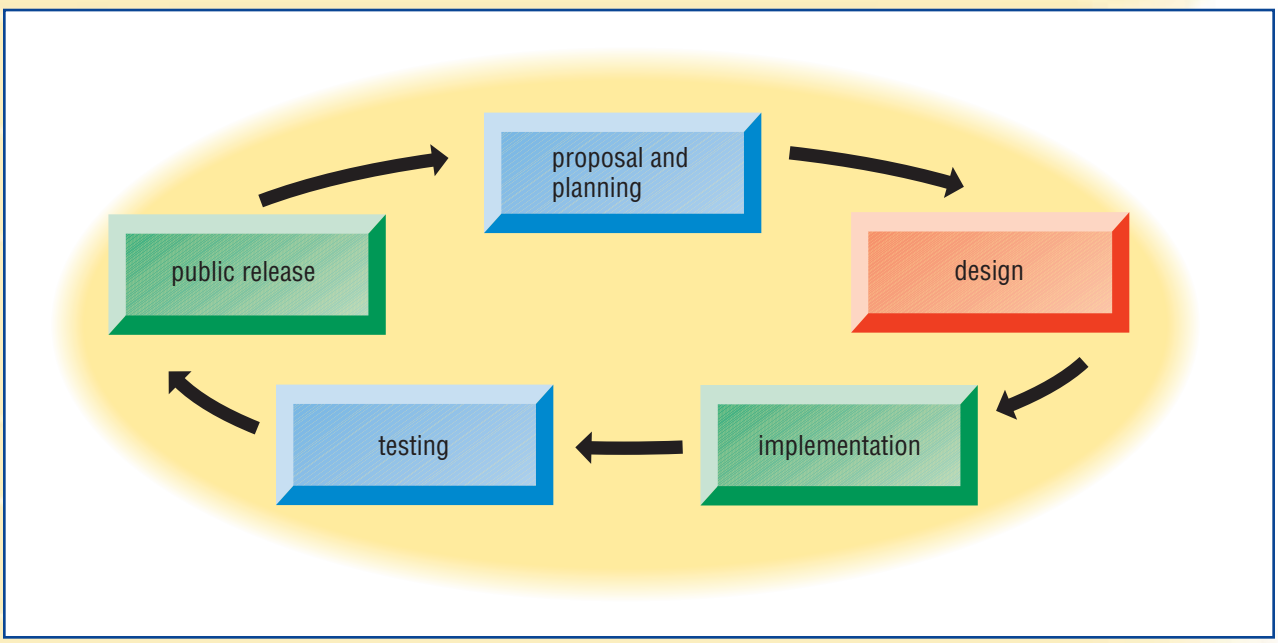
____ 8. A language created by Microsoft that is user-friendly, interpreted, and often used to quickly create graphic interfaces.

____ 9. One way to make decisions in a program.

____ 10. A set of techniques designed to speed up the process of software development.

Technology Illustrated > Identify the Process

What process is illustrated in this drawing? Identify the process and write a paragraph describing it.



Knowledge Check > Multiple Choice

Additional quiz questions are available on the Encore CD that accompanies this book as well as on the Internet Resource Center for this title at www.emcp.net/CUT3e.

Circle the letter of the best answer from those provided.

1. _____ can be reused in other programs.
   a. Variables
   b. Flowcharts
   c. Code modules
   d. Batch files

2. A(n) _____ is a set of instructions that tell a computer how to perform a particular task.
   a. input device
   b. output device
   c. control unit
   d. program

3. A program that translates a series of instructions written in a high-level language into low-level instructions that the computer can perform is known as
   a. source code.
   b. a compiler.
   c. object code.
   d. a debugger.

4. Which of the following was developed as a learning language?
   a. COBOL
   b. BASIC
   c. assembly
   d. C++

5. Which of the following is a low-level language?
   a. Visual BASIC
   b. C++
   c. machine code
   d. none of these

6. Which of the following skills is considered most important for a programmer?
   a. problem-solving skills
   b. language skills
   c. typing skills
   d. flowcharting skills

7. If a computer programmer calls someone a "code hacker," it means that person
   a. is a master programmer.
   b. doesn't program very carefully.
   c. breaks into computer security systems.
   d. is very organized.

8. Source code that is broken into sections for easy debugging and reusability is called _____ source code.
   a. modular
   b. low-level
   c. script
   d. compiled

9. What is the next step after thinking about a problem and coming up with a possible way to solve it?
   a. Try to write an algorithm for it.
   b. Write the program for it.
   c. Think of as many other solutions as you can for it.
   d. Try to state the problem clearly.

10. When programmers break a big problem into small segments to be completed one at a time, they are using the _____ approach.
    a. analyze-and-algorithm
    b. divide-and-conquer
    c. push-and-pull
    d. program-and-test

Things That Think > Brainstorming New Uses

In groups or individually, contemplate the following questions and develop as many answers as you can.

1. Do you think that cars will be able to drive themselves some day? What different parts of the car would a computer have to control? What different tasks would have to be performed? What impact would this kind of technology have on our daily lives?
2. If computers were capable of programming themselves, how would you tell them what you wanted them to do? Think of ways that a computer attempting to write a piece of software for you might extract the information it would need to figure out exactly what you wanted. Outline a step-by-step process that a future intelligent computer could follow to create new software. Do you think this type of technology would allow computers to take over from humans some day?

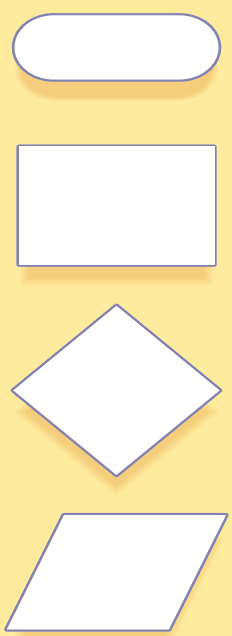
Key Principles > Completion

Fill in the blanks with the appropriate words or phrases.

1. The _____ language was originally designed to be an English-like language for handling database processing.
2. Errors in a program are called _____.
3. _____ is the process of removing errors from a program.
4. _____ is the act of writing a program.
5. A business language that specializes in report generation is called _____.
6. _____ is a new version of a popular programming language developed by Microsoft.
7. A program that stops running is said to have _____.
8. A common type of programming error that makes a program do the same thing over and over again, without stopping, is called a(n) _____.
9. In an effort to put rules into place concerning how to properly write a program, _____ was introduced in the early days of software development.
10. An incorrect instruction stated in correct syntax is called a(n) _____.

Tech Architecture > Label the Drawing

What does each of these symbols mean to programmers?



Techno Literacy > Research and Writing

Develop appropriate written responses based on your research for each item.

1. Is programming just for professionals? As computers permeate our society, many of our interactions with computer-operated devices are actually programming tasks, even though we may not think of them as such. Common activities that involve programming include setting up a VCR to record a show, or creating a worksheet in a spreadsheet program. How many other everyday programming examples can you think of? Create a list of the ones you find.
2. Does your school employ programmers? Many people are under the false impression that computer programmers work only for companies that create software. In reality, almost any large organization may employ programmers to do specialized work on their computer systems. Find out if your school has any programmers on its staff. What sort of work do they do? What languages do they use?
3. Could a bit of training as a programmer help your career? Students commonly think that they must dedicate their lives to programming in order to be successful programmers. In truth, most programming jobs do not involve pure programming. The majority of jobs require a mix of skills. Being able to handle minor programming tasks, such as the building of a simple Web page, is a useful skill for almost any career. What types of jobs could you do if you had some programming skills training? What career doors might open for you in the future?
4. Investigate the subject of macros as a form of programming. Which elements in the procedure of creating a macro are similar to the steps in developing a program? How are the two processes different? Using a program such as Word 2002 or Word 2003, create two macros to automate standard tasks you perform in writing research papers or in completing other types of assignments for college courses. Then write a summary of your research, including a description of the macros you created.

Technology Issues > Team Problem-Solving

In groups, solve each of the following problems using the six problem-solving steps outlined in this chapter. Number the steps one through six, and write a description of your efforts at each step.

Individual Issues

1. Starting a car.
2. Making a sandwich.
3. Deciding what to do for entertainment tonight.
4. Installing a modem in a computer.

Global Issues

1. Stopping an asteroid from colliding with the earth.
2. Providing fast Internet connections and making them accessible to everyone in the world.

Mining Data > Internet Research and Reporting

Conduct Internet searches to find the information described in the activities below. Write a brief report summarizing your research results. Be sure to document your sources using the MLA format (see Chapter 1, page 42, to review MLA style guidelines).

1. The Windows operating systems are the largest software projects ever completed in history. Find out what language Microsoft used to create its various products. Were they all created using the same language? How many lines of source code did the latest Windows operating system take to program? How many people worked on the project? How many years did it take to complete?
2. Software is already controlling many objects in our daily lives. Airplane autopilots are an obvious example of an intelligent program that performs a task normally associated with human expertise. How much of the time is a typical airliner under the control of a computer rather than a human? Do all major airliners use autopilots? Can autopilots land, take off, and fly a plane without any direction from human pilots?
3. Research rapid application development (RAD). What types of tools and techniques for writing software more quickly are included in RAD? How successful have RAD concepts been? What kind of improvement in development speed have users of these techniques achieved?

Technology Timeline > Predicting Next Steps

Computer language development has seen much progress during the course of its history. Recent trends appear to lead toward "super-languages" that combine features of low-level and high-level languages, and operate as either compiled or interpreted languages. Look at the timeline below, and predict developments in programming languages to the year 2020.

1946 Coding of the world's first computer (ENIAC) is performed in machine code only

1960 COBOL, a high-level compiled language, is invented by Grace Hopper

1964 BASIC, a high-level interpreted language, is released

1971 C, a compiled language that is lower level and more machinelike than most previous languages, is released

1980 Object-oriented programming languages are developed, raising the level of modularity and making the programs more abstract (higher level)

1995 Java is released as a slightly higher-level version of C that runs in an interpreted fashion on the Internet

1998 Microsoft releases Visual J++ as a competitor to the industry-leading Java from Sun Microsystems

1998 The XML Working Group officially releases the first version of the XML special-purpose markup language (XML 1.0)

2000 Microsoft releases C# as part of the Visual Studio development toolkit

2001 Eclipse project begins

2002 Microsoft releases VB.NET, Visual C#, and ASP.NET for Web development under the .NET framework

2006 XML 1.0 Fourth Edition and XML 1.1 Second Edition are released.

Ethical Dilemmas > Group Discussion and Debate

As a class or within an assigned group, discuss the following ethical dilemma.

With the widespread use of the Internet, it is much easier for programmers to access previously written code modules or objects to shorten development time. Some code is viewable by anyone with a Web browser using the "View Source" browser function. In addition, a common practice in the programming community is to share code freely, posting it on programming bulletin boards and special-interest Web sites. When programmers share code from an application they are developing at work, they are likely giving away intellectual property of the company. When programmers find code on the Internet to use in an application at work, they may be unknowingly using the intellectual property of another company. Are these practices unethical? Should programmers be required to identify the source of any programming code they have not written themselves? Would you think less of the programmer's skill if programming code is "copied and pasted" from another source?